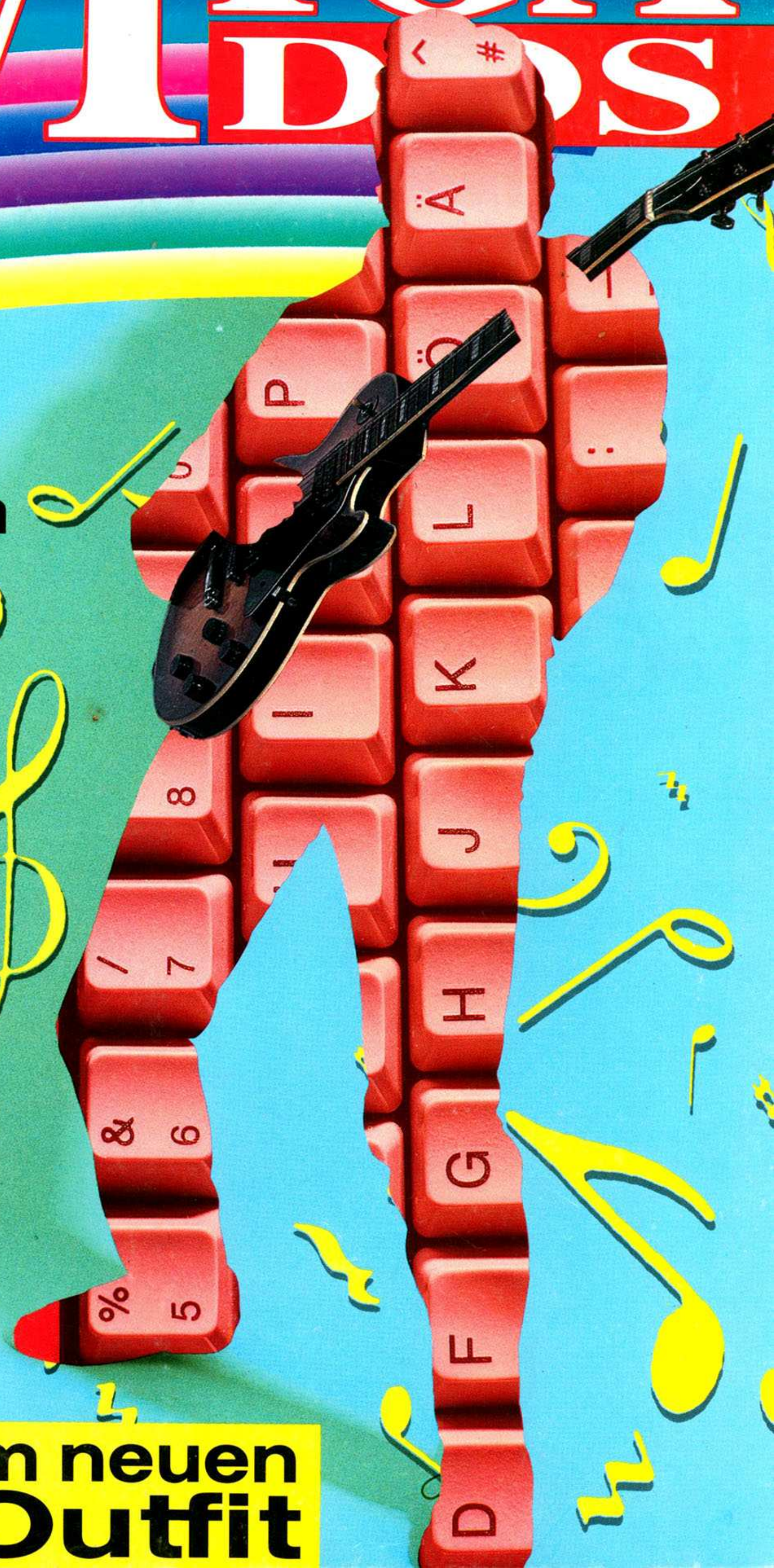
AMIGA DOS

DAS MAGAZIN FÜR AKTIVE AMIGA-ANWENDER

DM 6,50
sfr 6,50
öS 52

AMIGA DOS

8/90



8/90
ISSN 0937-2717
DMV-Verlag

Tips und Tricks

Light-Show am Druckerport

Wettbewerb

– Super LCD-Spiele zu gewinnen

Alles über Musik

– Hardware
– Software
– Tips von Profis

A500 im neuen Outfit



Wetten, daß...

... **W**olfgang Amadeus Mozart, würde er heute leben, einen Amiga auf seinem Schreibtisch stehen hätte? Die kleine Nachtmusik, komponiert mit vierstimmigem Digitalsound – das wäre doch was. Armer Wolfgang Amadeus, konnte er doch nie die Klangqualitäten der heutigen Zeit für seine revolutionären Ideen im Musikbereich nutzen.

"Musik mit richtigen, echten Instrumenten gespielt, das ist wahre Musik". Wer das heute noch glaubt, geht irgendwie nicht mit der Zeit. Komponisten, Arrangeure, Musiker, fast keiner kommt heute mehr ohne den Einsatz eines Computers aus. Und was den Profis recht ist, kann den Amateuren nur billig sein. Wozu hat man einen der besten "Musiker" im Computerbereich zu Hause stehen? Piepsen können viele, Wellenformerzeugung haben einige auf ihrem Arbeitsplan, aber digitalisierte Sounds zu fetziger Rockmusik vereinen, das kann der Amiga am besten.

Und somit haben wir in diesem Monat die Ausgabe der AMIGA DOS unter das Motto "Musik und Amiga" gestellt. Wir bringen Ihnen Grundlagen der Soundprogrammierung genauso nahe, wie wir Ihnen kommerzielle Programme vorstellen, die wohl für jeden Geschmack etwas beinhalten.

Tests über neue Sound-Editoren, MIDI-Sequencer und Sound-Sampler werden auch weiterhin ihren Teil im Heft zugestanden bekommen, schließlich ist unser Amiga nicht umsonst ein Soundgenie.

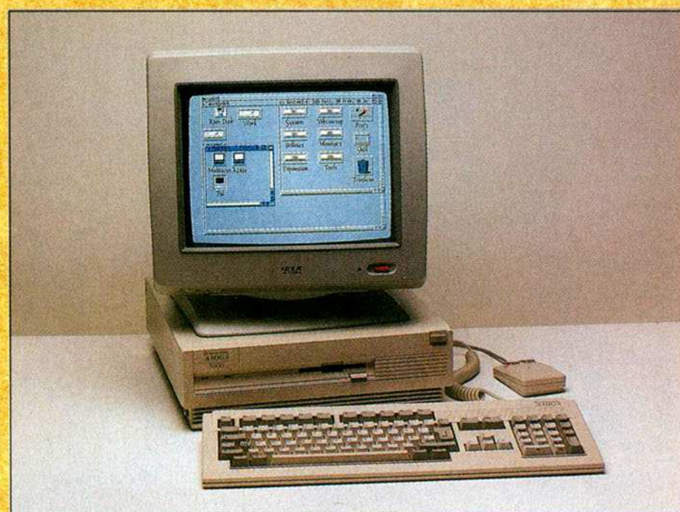
Na also – play it again, Amiga.

Herzlichst, Ihr

Jürgen Borngießer

Jürgen Borngießer

INHALT



Der Amiga 3000, Commodores neues Flaggschiff – was leistet er wirklich? Wird er nur als Traumcomputer angepriesen, oder ist er es?

Seite 18



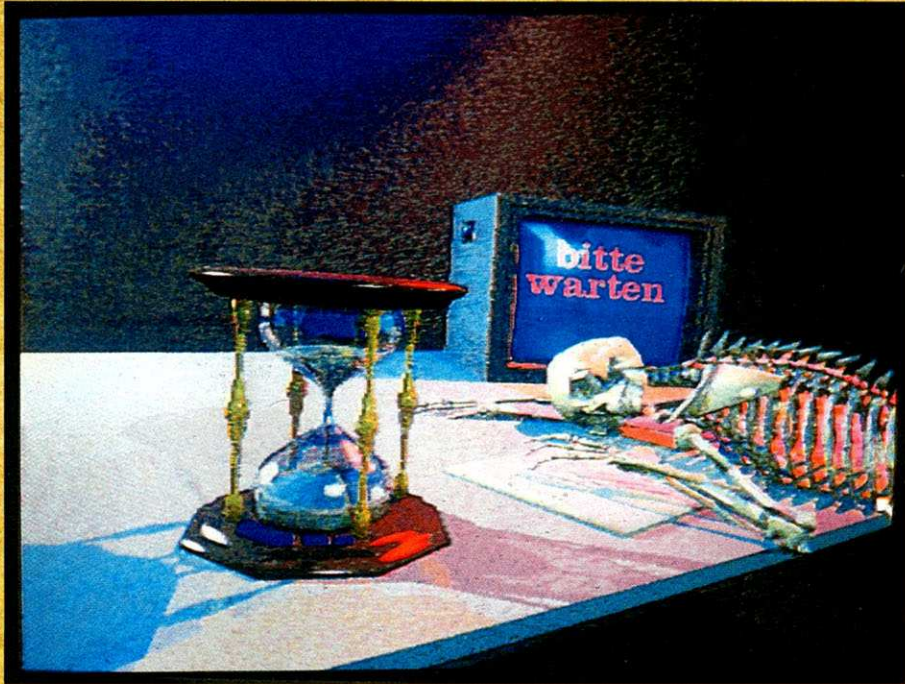
Wie kommt bloß die Musik aus dem Amiga? Rund um PAULA – den Soundchip – und seine Programmierung dreht sich unser Titelthema "Musik"

Seite 26



»Sokoban« war eines der Highlights auf dem PC. Ein pfiffiger Programmierer hat mit »Hard-Work« dieses Spiel für den Amiga umgesetzt. Näheres zum Kistenschieben lesen Sie auf

Seite 79



Raytracing erzeugt faszinierende Bilder. Wie's funktioniert, erfahren Sie auf

Seite 91



»Cloud Kingdoms«, ein neuer Top-Hit unter dem Label Millenium, entführt Sie in magische Königreiche

Seite 113

LISTING

WERKSTATT

PUBLIC DOMAIN

SPIELE

RUBRIKEN



Marshal M. Rosenthal

World of Amiga

Eine Multimediashow in New York

Es hätte kaum schlimmer kommen können - dieser Gedanke drängt sich unwillkürlich auf, wenn man versucht, sich seinen Weg durch die geschäftigen Menschenmassen in New York zu bahnen. Unser Ziel ist die Ausstellung World of Amiga.

Diese Ausstellung ist nicht in einem Convention Center oder einem ähnlich zivilisierten Ort gelegen, sondern in einem Gebäude an Pier 91. Das hört sich natürlich nach einem malerischen und romantischen Platz für eine Computershow an, aber man sollte auch folgendes in Betracht ziehen: Die einzige Möglichkeit, diesen Platz zu erreichen, ist zu Fuß zu gehen oder ein Taxi zu nehmen. Aber die Taxifahrer sind auch nicht besonders daran interessiert, einen Fahrgast an einem Ort aussteigen zu lassen, wo es schwierig ist, neue Kundschaft zu bekommen.

Zu Fuß kreuzen wir die zehnte Avenue und erreichen das Ufer, das längs der 58. Strasse verläuft. Von dort schlägt uns bereits ein betörender Geruch entgegen, da hier die Firma ihren Standort hat, die die Kanalisationsrohre in New York ersetzt. Ergo beschleunigen wir unsere Schritte und betreten den Fahrstuhl, der uns in die "World of Amiga" befördert.

Obwohl es erst Freitagnachmittag ist, ist die Show überfüllt. Als erstes springt der große Commodore-Stand ins Auge, auf dem sich eine Menge Programmierer und Entwickler ein Stelldichein geben. Das Kronjuwel ist der Amiga 3000, der hier das erste Mal seit seiner Vorstellung vor einer Woche wieder zu sehen ist. Einen Bericht zum Amiga 3000 finden Sie an anderer Stelle in diesem Heft. Dort auf der Show versucht natürlich jeder, diesen Supercomputer einmal anzufassen und das neue Betriebssystem auszuprobieren.

Der Toaster – lang angekündigt – jetzt erhältlich

Ein weiterer Star dieser Ausstellung ist der Videotoaster von Newtek. Die Menschenmassen formen einen durchgehenden Wall zwischen dem



Bild 1. Der Videotoaster bietet zahlreiche Kontrollmöglichkeiten während der Arbeit Photography by Marshal M. Rosenthal



Bild 2. Die Steckkarte mit ihren Videoein- und -ausgängen bildet die Hardware des Toasters Photography by Marshal M. Rosenthal

A3000 und dem Toaster, es ist so gut wie unmöglich, sich hier einen Weg zu bahnen. Nun arbeitet Newtek ja bereits seit vier Jahren an diesem Produkt, und auf jeder Messe war bisher zu hören, daß der Toaster spätestens zur nächsten Messe endgültig fertig sein soll. Aber nun ist der Knoten geplatzt, der Toaster ist fertig und wird offiziell zum Kauf angeboten. Und es ist schon beeindruckend, was der Toaster zusammen mit den Möglichkeiten der Multitasking-Maschine Amiga zu leisten vermag.

Software-gesteuert, besteht der Toaster aus einer Steckkarte, die in einen Amiga 2000, 2500, und A3000 eingesetzt werden kann. Notwendig zum Betrieb sind ein Arbeitsspeicher von 3 MByte.

Er enthält ein eigenes Set von Custom Chips, die dem Toaster zu einer Fernsehbildqualität verhelfen. Und das sowohl in der Auflösung als auch im Hinblick auf die Farbvielfalt: 16,8 Millionen Farben sind schon eine Leistung. Die Software, die den Toaster steuert, ist auf vollen Mausbetrieb ausgelegt. Die Ein- und Ausgabebuchsen sind direkt auf der Karte untergebracht; wir finden vier Eingänge für externe Videosignale, zwei digitale Eingänge und einen Matte-Generator. Ausgestattet mit einem doppelten Frame-Buffer, ist der Toaster in der Lage, 3D-Images (24Bit) wiederzugeben, eine optionale Grafik-Software, die demnächst herauskommen soll, ermöglicht dann die Nachbearbeitung.

Ein spezieller Schalter ermöglicht Effekte wie Zerbröseln, Wischeffekte, Szenen- und Farbeffekte. Mit Hilfe des Preview Outputs kann man sich diese Dinge auf einem kleinen Kontrollmonitor schon ansehen, während sie durchgeführt werden. Alle digitalen Videoeffekte arbeiten in Echtzeit, das Bild kann damit gedreht, verzogen und hereingezoomt werden, was allerdings nur einen kleinen Einblick in die tatsächlichen Möglichkeiten des Toasters gibt. Außerdem gibt es eine Farbverarbeitung, mit der sich verschiedene Filtereffekte erzielen lassen, die uns aus der Fotografie bekannt sind. Ergänzt wird dies durch einen Zeichengenerator, der in der Lage ist, erstellte Texte zu scrollen und über zahlreiche Fonts verfügt.

Interessant ist auch der Preis. Der Toaster kostet in den Vereinigten Staaten 1595,– US$. Und das ist bestimmt nicht zuviel Geld für eine so leistungsfähige Karte.

Natürlich stellen noch zahlreiche andere Hersteller ihre neueste Software aus. Überraschend war, daß auch Electronic Arts mit einem Stand auf der Show erschienen.

Spiele-Software ohne Ende

Neben der Vorstellung von »DeLuxeVideo III« waren noch drei Labels von EA auf dem Stand beherbergt. Lucasfilm zeigte »LOOM«, das wir Ihnen bereits vorgestellt haben. Außerdem war »Battle of Britain« mit verbesserter Grafik zu sehen, die die Fähigkeiten des Amiga voll ausnutzt. Bild 3 zeigt einen ersten Eindruck der neuen Grafik.

UBI-Soft war auch auf der Show vertreten und zeigte mehrere ältere Spiele, die allerdings nie offiziell in den USA erschienen sind, zum Beispiel »Puffy's Saga«. Das



Bild 3. Verbesserte Grafiken bei »The Battle of Britain«



Bild 4. »LOOM« bietet eine große Anzahl sehr gut gelungener Grafiken



Bild 5. »Unreal«, das neue Arcadenspiel von UBI-Soft



Bild 6. David Jones erklärt den »Showmaster«

interessanteste Produkt in diesem Zusammenhang ist sicher »Unreal«. Es handelt sich um ein Arcadenspiel mit ein klein wenig Rollenspiel gewürzt und besticht durch faszinierende Grafiken und eine Unzahl von Finsterlingen.

Miles Computing hat sich übrigens Electronic Arts angeschlossen, was ein Vorteil für alle Amiga-User sein dürfte. Ihr neuestes Produkt, »Fool's Errand«, ist ein ungewöhnliches Spiel, das auf Rätseln basiert. Es ist schon eine Menge Logik und Grübeln erforderlich, um erfolgreich das Königreich zu retten.

Viel mehr Arcade-orientiert ist da »Aquanaut«, ein Adventure unter Wasser.

Hologramaphone Research arbeitet auch an zwei neuen Programmen. Da wäre zunächst »Kepler«, ein neues Musikprogramm für den Amiga. Außerdem soll demnächst »DNA« vorgestellt werden, ein Programm, das Kunst erzeugen kann, wenn man den Angaben der Hersteller Glauben schenken darf. Ebenfalls aus diesem Hause steht uns in naher Zukunft »Delphic Wells« bevor. Dieses Programm soll die Zukunft voraussagen können. (Kann wohl nicht schlimmer sein als die tägliche Wettervorhersage.) Es hört sich schon ein bißchen bizarr an; wir werden Ihnen dieses Programm vorstellen, sobald es verfügbar ist.

»Showmaker« von Gold Disc

Spezielle Veranstaltungen sind natürlich auch Bestandteil der Show. Gold Disc konnte hier besonders mit seinem »Showmaker« von sich reden machen. In der Hospitality Suite von Gold Disc hatten wir die Möglichkeit zu einem Gespräch mit dem Direktor David Jones.

David Jones: *"Multimedia ist die Computerphrase der 80er Jahre, aber erst jetzt, in den 90ern, wird es wirklich möglich. Showmaker versucht, den Anspruch des echten Multimedia-Desktop-Videos zu verwirklichen. Dieses Programm erlaubt die Integration von Animation, Grafik, MIDI, Computer-Sound, Laser-Discs, softwaregesteuerten Genlocks und vielen weiteren Videoquellen. Showmaker wurde konzipiert, um professionellen Ansprüchen zu genügen, aber auf einer Basis, die auch für den Normalverbraucher erschwinglich ist. Showmaker verfügt auch über einen Titelgenerator und eine Vielzahl von Videoeffekten, wie Wischeffekte und Ein- und Ausblenden eines Bildes. Eines der wichtigsten Features von Showmaker ist seine Fähigkeit, Musik automatisch zum Video zu synchronisieren. Damit gehört ein wichtiges Problem von Desktop-Video der Vergangenheit an. Denn jetzt können Sie beispielsweise eine Animation in Frames pro Musiktakt statt Frames pro Sekunde einstellen."*

David Jones führt uns den Showmaker vor, indem er ein kurzes Stück "Multimedia" zusammenstellt. Er benutzt dazu stehende IFF-Grafiken, ein Stück Animation und Musik von einem MIDI-Keyboard. Ein sehr einfaches und intuitives Interface ermöglicht es, alle Bestandteile auf einfachstem Wege zusammenzusetzen. Das Programm ist außerdem in der Lage, Storyboards anzusehen und auch auszudrucken, auch dann, wenn einige Blätter des Storyboards noch in Arbeit sind. Interessant ist auch, daß ein externer Clock-Eingang für MIDI vorhanden ist.

Neues auch vom Zeitschriftenmarkt

Eine neue Zeitschrift gibt es jetzt speziell für die Amiga-Fans, die ein Bridgeboard ihr eigen nennen und die MS-DOS-Welt über Ihren Amiga kennenlernen wollen. Recht informativ werden PC-Infos vermittelt. Aber auch Tips und Tricks kommen nicht zu kurz. Sollten Sie Interesse an diesem Magazin haben, können wir Ihnen eine interessante Information bereitstellen: AMIGA-DOS-Leser können das Magazin zu einem reduzierten Preis abonnieren. Nähere Informationen entnehmen Sie bitte der Box.

Handliche Scanner

Zurück auf der eigentlichen Messe, wenden wir uns noch einmal Migraph zu, die für ihre Hand-Scanner bekannt sind. Neu umgesetzt wurde »Touchup« für den Amiga. Dies ist ein hochauflösendes Schwarzweißprogramm, um Änderungen an importierten Bildern vorzunehmen. Spezielle Merkmale sind die Zoomfähigkeit sowie einige Zeichenhilfen und die Möglichkeit der Grafikmanipulation. Außerdem können Files vom Atari ST, MS-DOS-Maschinen sowie Macintosh-Computern eingelesen werden. Ebenfalls enthalten ist ein hochauflösender Druckertreiber, mit dem die Bilder anschließend zu Papier gebracht werden können. Das Programm wird allerdings nur zusammen mit dem Scanner verkauft, der vier verschiedene Auflösungen anbietet.

(mm)

AMIGA DOS Info

Amiga Crossings
Deland Editorial Services
345 East 93rd Street, #26E
New York, NY 10128

Ein Überseeabonnement (1 Jahr/12 Ausgaben) kostet für AMIGA-DOS-Leser inklusiv Postgebühren nur 36,- US$ statt normal 46,- US$.

Gold Disc
2175 Dunwin Drive, #6
Mississaugua, Ontario
Canada L5L, 1X2

Electronic Arts
1820 Gateway Drive
San Mateo, California 94404

Hologramaphone Research
6225 SW 145th Street
Miami, Florida 33158

Migraph
200 South 333rd Street, Suite 220, Federal Way, Washington 98003

NewTek
115 West Crane
Topeka, Kansas 66603

Es lebe die Gemütlichkeit

Amiga '90 Basel

Sehr groß angekündigt war sie ja, die Amiga-Messe in Basel – Aussteller, Fachpresse und Zuschauer durften gespannt sein, was sie wohl erwarten würde.



Bild 1. Vortex Athlet im Amiga 3000

Das AMIGA-DOS-Team war exklusiv für Sie live dabei. Was war nun das Besondere an dieser Amiga-Messe? Zunächst einmal muß man kritisch anmerken, daß diese Messe wohl für Aussteller, Besucher und die schreibende Zunft ein großer Flop war. In einer einzigen Halle konnte man die Zahl der Aussteller fast an zwei Händen abzählen. Der Zuspruch war mehr als dürftig. Den Ausstellern, die mit ihren Ständen vertreten waren, muß man jedoch ein großes Lob zollen. Trotz widriger Umstände, wie übertriebenem Formalismus etc., präsentierten sie viele neue Produkte.

Natürlich wurde das neue Flagschiff von Commodore, der Amiga 3000, dem Publikum vorgestellt. Mit recht beeindruckenden Präsentationen wurden die neuen Fähigkeiten dieses Gerätes, wie Grafik, neue Benutzeroberfläche und natürlich die rasante Geschwindigkeit, hervorgehoben. Im Testteil dieser Ausgabe finden Sie einen ausführlichen Praxistest über den Amiga 3000.

Die Firma Vortex präsentierte auf ihrem Stand die gesamte Produktpalette für den Amiga: Floppy-Stationen, Speichererweiterungen und Festplatten-Subsysteme für den Amiga 500, Amiga 1000 und den Amiga 2000. Ein besonderes Augenmerk verdient die Tatsache, daß auf dem Vortex-Stand auch ein Amiga 3000 aufgebaut war, in dem die Einsteckfestplatte Vortex Athlet eingebaut war. Man konnte beobachten, wie diese Festplatte ohne große Probleme mit dem Amiga 3000 zusammenarbeitete. Vortex Athlet verfügt über Kapazitäten von 40 MByte bis zu 180 MByte und ist in erster Linie für den Amiga 2000 vorgesehen.

Die amerikanische Firma GVP war mit einem eigenen Stand vertreten und stellte ihre neuesten Produkte vor. Die Produktpalette reicht von Festplatten mit Fast-RAM für den A500 über schnelle Hardware-Lösungen bis hin zum superschnellen Turboboard in 68030-Technologie. Der 68030-Prozessor mit 50 MHz im Impact-A3001-Turbokit stellt natürlich das absolute Nonplusultra dar. Den deutschen Vertrieb hat die Firma DTM in Wiesbaden.

»Bars&Pipes« nennt sich ein neues MIDI-Music-System, das von der amerikanischen Firma Blue Ribbon Bakery vorgestellt wurde. Überzeugend ist die Leistungsfähigkeit dieses Programms, das fast alles bisher Dagewesene in den Schatten stellen könnte. »Bars&Pipes« verfügt über ein eigenes System von "Leitungen" zum Verbinden der verschiedenen Spuren und Bearbeitungsmodule, die den Fluß der MIDI-Daten von der Eingabe bis zur Ausgabe bestimmen. Wir werden Ihnen in einer der nächsten Ausgaben einen ausführlichen Testbericht liefern.

Am gleichen Stand wurde ein modulares System mit dem Namen »Pi« vorgestellt. »Pi« ist eine Programmserie, die im besonderen für Schüler und Studenten geeignet ist. Insgesamt fünf Module sorgen dafür, daß der Umgang mit Mathematik kaum noch Probleme bereitet. »Pi-Plotter« hilft bei der grafischen Umsetzung von Funktionen, »Pi-Matrix« ist ein exzellenter Matrizenrechner. Ein wissenschaftlicher Taschenrechner und Einheitskonverter, ein Modul für lineare Optimierung und ein Statistik-Modul komplettieren ein rundum gelungenes Paket, das von der Firma DTM vertrieben wird.

Das langangekündigte MW-500 wurde von der Firma Miky Wenngatz im Einsatz vorgestellt. Das MW-500 ist ein spezieller Umbausatz für den A500. Einen ausgiebigen Testbericht finden Sie im Testteil dieser Ausgabe.

»Turbo Print Professional« ist eine logische Weiterentwicklung des bewährten Produkts »Turbo Print«, das von der Firma Irsee Soft vorgestellt wurde. Viele neue Features erleichtern die Arbeit mit dem Drucker allgemein. Die Ausdrucke sprechen für sich.



Bild 2. Das 2,5-Zoll-Laufwerk im Einsatz



Bild 3. »Bars&Pipes«, ein neuartiges Musiksystem aus den USA

In der nächsten Ausgabe finden Sie einen ausführlichen Testbericht.

»Colourbox« nennt sich ein neuartiges Videomisch- und Effektsystem für den A500, A2000 und A2500, das von der Firma Intelligent Memory präsentiert wurde. Das Colourbox-System ist ein 6-Regler-Mischpult, mit dem sich alle beliebigen Farben mischen lassen. Damit ist es möglich, nahezu jeden Farbbereich des Videosignals einzugrenzen und durch das Computersignal zu ersetzen. Dieses System ist nach Angaben von Geschäftsführer Wolf Dietrich voraussichtlich ab Oktober dieses Jahres erhältlich. Der Preis wird bei 1800 DM liegen.

Die neueste Version 3.3 ihres Modula2-Compilers stellte die Firma A + L AG vor. Daneben wurden mit »M2Optimize« und »StonEd« zwei weitere nützliche Module präsentiert. Mit Hilfe von »M2Optimize« werden die zum Linken bereiten Objektdateien auf unbenutzte Referenzen analysiert und diese dann auch gelöscht. Damit wird erheblich Platz eingespart. »StonEd« ist ein Alternativ-Editor zum standardmäßig mitgelieferten »M2Emacs«. Er ist in erster Linie für Programmierer entwickelt worden und arbeitet voll mausunterstützt.

Die Firma Merkens EDV stellte in Zusammenarbeit mit der Firma Videocomp ein neues System im Bereich Digitalisierung vor: das VD 2001. Dieses System arbeitet mit einer 24-Bit-Farbauflösung, das bedeutet, daß 16,8 Millionen Farben dargestellt werden können. Es verfügt über über einen eigenen RGB-Ausgang. Bilder können in rund 20 Millisekunden eingefroren werden. Raytracing-Bilder lassen sich ohne Probleme einladen. Daneben verfügt das VD 2001 über eine eigene Maskenbitplane, mit der sich eine Vielzahl von Effekten erzielen lassen und eine AREXX-Schnittstelle. Das System ist auf allen Amigas lauffähig, einschließlich des A3000. Der Preis wird sich um 3000 DM bewegen.

3-State Computertechnik wartet ebenfalls mit einigen neuen Produkten auf. Neben Autoboot-Harddisks für alle Amigas, internen Speichererweiterungen für den A500 und Floppy-Laufwerken wurde mit der Megamix 2000 eine



Bild 4. Special effects mit der Colourbox erzielt



Bild 5. Merkens EDV präsentiert das VD 2001



Bild 6. Neue Hardware von der Firma FSE

neue RAM-Karte für den A2000 vorgestellt. Eine wichtige Information für alle interessierten User: Medusa, der ST-Emulator, wird nicht mehr von der Firma Combitec vertrieben, sondern von 3-State.

Die Firma M.A.S.T. war auf der Amiga-Messe Basel mit einem Stand vertreten. Neben bereits bewährten Produkten wurden auch einige Neuheiten vorgestellt. Präsentiert wurden neben dem Festplattenlaufwerk Fireball für A2000, das ab 45 MByte mit einem zusätzlichen Cache-Speicher ausgerüstet ist, noch das externe Festplattenlaufwerk Tiny Tiger für A500/1000/2000. Ein besonderes Bonbon wurde mit »Blitzbasic« vorgestellt. Dieses neue BASIC erzeugt sehr schnellen Source-Code und verfügt noch über einen eingebauten Compiler. Der Preis für Blitzbasic dürfte bei rund 150 DM liegen.

Ein funkelnagelneues Produkt stellte die Firma Frank Strauß Electronic vor: eine AT-Bus-Platte für den A500. Diese Hardware-Lösung überzeugt bereits durch ihre hohe Geschwindigkeit, die den A500 zu einem Sprinter werden läßt. Wir werden in einer der nächsten Ausgaben einen Testbericht vorlegen. Eine weitere Neuerung wurde mit der Quantum-Festplatte für den A500/1000 präsentiert. Das besondere daran ist, daß diese Festplatte über ein 16-Bit-Interface verfügt. Als Treiber-Software wird das neue »BOIL3« eingesetzt.

Das langerwartete 2,5-Zoll-Laufwerk wurde von der Firma Gigatron erstmals auf der Baseler Amiga-Messe vorgestellt. Die geringen Ausmaße und die gute Verarbeitung dieses Laufwerks lassen darauf schließen, daß der Firma Gigatron wieder ein gutes Produkt gelungen ist. Wir werden Ihnen in einer der nächsten Ausgaben einen ausführlichen Testbericht liefern.

Die Firma Kupke aus Dortmund präsentierte auch einige Neuheiten auf der Messe. Der Stand war ständig umlagert. Grund dafür war eine Pistole, mit der man auf den Bildschirm schießen konnte. Entsprechende Spiel-Software gab es ebenfalls dazu. Auch eine kleine Digitalisierungsstation für zu Hause wurde präsentiert. Ein neuer Streamer, der sich über ein recht gutes und neuartiges Backup-Programm ansteuern läßt, wußte zu überzeugen.

Zum Abschluß möchte ich noch eine Prognose wagen: Nach diesem "Flop" dürfte die Amiga-Messe Basel wohl gestorben sein. Man darf gespannt sein, was seitens der Aussteller für die Schweizer Messezukunft ins Auge gefaßt wird.

(vb)

News von UGA-Software

United Graphic Artists, kurz UGA genannt, präsentiert neue Low-Budget-Software. »The Musical Enlightenment« ist ein recht guter Musik-Editor zum kleinen Preis. Das Programm bietet vier Teile voller leistungsfähiger Menüpunkte, um Lieder zu editieren, Samples zu bearbeiten und besondere Effekte zu erzeugen. Mindestvoraussetzung ist ein Amiga mit 512 kByte Speicher und einem Laufwerk. Das Programm verfügt über ein einfach zu bedienendes Benutzer-Interface. Mit Hilfe eines Effektgenerators können IFF-Samples in beeindruckender Weise zu neuen Soundeffekten umgeändert werden. Der Preis von »The Musical Enlightenment« beträgt 59,– DM. Mit »PowerPacker Professional« stellt sich ein sehr mächtiges Utility vor. Die neueste Version ist in der Lage, 40- bis 50mal schneller zu crunchen als die Vorgängerversionen. Daneben verfügt sie über eine Vielzahl neuer Features, wie cryptografierte Files, neue Kommandos, wie »Skip crunched files« oder »Delete source files«, einen verbesserten File-Requester und vieles andere mehr. Der Preis für Power Packer Professional beträgt 39,– DM.

Info:
UGA Distribution Deutschland
Im Mühlenfeld 11
4650 Gelsenkirchen
Tel: 0209/203911



Ein nützliches Utility – Power Packer Professional

Neue Elsa-Modems mit Fax

Rechtzeitig zur Cebit erteilte das Zentralamt für Zulassungen im Fernmeldewesen (ZZF) dem Aachener Modemhersteller Elsa GmbH eine Reihe von Zulassungen, die den Betrieb der bekannten Microlink-Modems mit Sendfax-Option ermöglichen. Die Geräte Microlink 9624T2X, 2400T2X und 2400PCX (X steht jeweils für Fax) können Dokumente vom PC direkt auf Telefax senden.

Die mitgelieferte Elsafax-Software kann Texte und Grafiken versenden, verwaltet mehrfache Anwahlverzeichnisse und ermöglicht Rundsenden an Teilnehmergruppen. Grafiken im PCX-Format können mit einfachen Befehlen in den zu versendenden Text eingebunden werden (zum Beispiel Einfügen eines Briefkopfes). Als Kommandozeilen-Version ermöglicht Elsafax die Übergabe von Sendelisten, die uhrzeitgesteuert abgearbeitet werden. Ein Sendejournal, ähnlich dem eines Fernkopierers, wird ebenfalls erzeugt.

Das ZZF hat Elsa die Zulassung ohne besondere Beschränkungen erteilt. Ein zusätzlicher Telefonanschluß oder ein separates Faxgerät sind daher zum Betrieb nicht erforderlich. Gegen einen Aufpreis von 350,– DM zuzüglich Mehrwertsteuer können folgende Modems mit Sendfax-Option ausgestattet werden: Microlink 9624T2, 2400T2 und 2400PC. Bereits ausgelieferte Geräte können nachgerüstet werden. Ein Katalog mit weiteren Informationen über die Modems kann bei Elsa angefordert werden.

Info:
Elsa GmbH
Sonnenweg 11
5100 Aachen
Tel: 0241/477 89 12
Fax: 0241/477 89 60

Neue Bücher

Der Verlag Gabriele Lechner aus München stellt in seiner Workshop-Reihe einige Neuheiten vor, zum einen die zweite überarbeitete Auflage von **Turbo Silver 3.0**, die auch **Turbo Silver 3.01 SV** enthält. Die Themen dieses Workshops reichen von Objektgenerierung, Kamerahandhabung bis hin zu den neuen Befehlen und Textures der SV-3.01-Version. Der Preis beträgt 69,– DM (inklusive einer Diskette).

Ein weiterer Workshop beschäftigt sich mit **Sculpt 3D/4D**. Hier bringen die Autoren C. Obermaier und W. Friedhuber eine Einführung in die komplexe Welt der 3D-Computergrafik. Sie entwerfen Modelle, erläutern die Bedienung des Tri-View und konstruieren komplexe Objekte, um nur einige Features zu nennen. Der Preis dieses Workshops beträgt 69,– DM (inklusive einer Diskette).

Eine konsequente Weiterentwicklung des Bestsellers **Abenteuer Computer** wird mit dem Buch **Deluxe Paint III Profitips** vorgestellt. Tips und Tricks im Umgang werden nicht nur beschrieben, sondern auch auf zwei Disketten dokumentiert. Die Spannbreite der Themen reicht von Fantasy-Illustrationen bis zu klassischen Trickfilmtechniken. **Deluxe Paint III Profitips** wird für 98,– DM (inklusive zwei Disketten) beim Verlag erhältlich sein.

Info:
Verlag Gabriele Lechner
Planegger Str. 1
8000 München 60

Infos zu DTP

Eine neue Broschüre über Desktop Publishing am Amiga stellt die Firma M.A.R. Computershop aus Österreich vor. Sie wurde auf einem Amiga 2500 DTP unter Beteiligung der Programme »Professional Page 1.31d«, »Professional Draw 1.0d«, »Advantage«, »Transscript« und »Word Perfect 4.1d« entwickelt. Die Broschüre soll Grundsatzwissen über Schrift, Gestaltung und Verwendung von DTP vermitteln. Den deutschen Vertrieb hat die Firma GTI übernommen.

Info:
GTI GmbH
Zimmersmühlenweg 73
6370 Oberursel
Tel: 06171/3048-9

Englisch perfekt beherrschen

Beim Erlernen einer Fremdsprache ist es nicht damit getan, einfach nur Vokabeln zu pauken. Vielmehr kommt es darauf an, den Zusammenhang zwischen den einzelnen Satzkomponenten zu verstehen. Denn wer würde schon den deutschen Satz: "Du bist schwer auf Draht" mit "You are heavy on wire" ins Englische übersetzen?

Man merkt sofort, daß es auf bestimmte Phrasen, also spracheigentümliche Redewendungen ankommt, will man Konversation in einer Fremdsprache betreiben. An dieser Stelle klinkt sich das Produkt Phrasen-Trainer der Firma Software 2000 ein. Im Gegensatz zu bisher üblichen Englisch-Lernprogrammen werden nicht nur Vokabeln, sondern ganze Sätze abgefragt, wahlweise vom Deutschen ins Englische und umgekehrt. Das Besondere an diesem Programm ist, daß mehrere Möglichkeiten der richtigen Beantwortung anerkannt werden, sofern sie sich in dem entsprechenden Datensatz befinden.

Der Phrasen-Trainer ist für knapp 60 DM im Fachhandel erhältlich.

Aktuelles vom Unterhaltungssektor

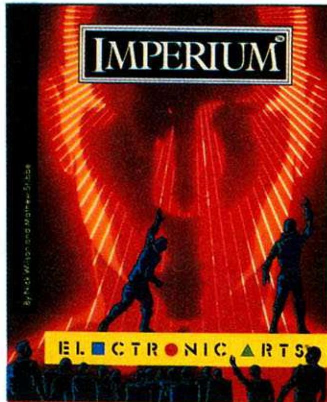
Die Firma **Electronic Arts** präsentiert mit »Imperium« ein neues Strategiespiel mit großer Komplexität und langanhaltender Faszination. Das Programm verfügt über 100 Welten, die man erobern und unterwerfen kann, daneben über ein großes Repertoire an musikalischer Untermalung. Das genaue Erscheinungsdatum lag bei Redaktionsschluß leider noch nicht vor.



Die Gralssuche auf dem Amiga – »Conquest of Camelot«

Die Firma **Sierra Online** hat die Amiga-Version des beliebten PC-Spiels »Conquest of Camelot« angekündigt. Begeben Sie sich mit den Rittern von König Artus Tafelrunde auf die Suche nach dem "heiligen Gral". Nach der attraktiven PC-Version darf man auf die Amiga-Version, die in Kürze erscheinen wird, gespannt sein.

Neuigkeiten von **The Disk Company**: Nachdem »Ports of Call« bereits vor zwei Jahren einen respektablen Erfolg auf dem Amiga verbuchen konnte, wird dieses geniale Spiel in Kürze in deutscher Sprache neu aufgelegt. Wenn Sie Spaß daran haben, mit Ihrer eigenen Reederei einen schwunghaften Handel zu betreiben und als Kapitän auf den sieben Weltmeeren umherzuschippern, dann dürfte dieses Spiel genau das richtige sein.



Ein neues Strategiespiel – »Imperium«

Leider ist der Preis mit 84,– DM immer noch etwas zu hoch angesetzt.

Endlich kommt er: »The Punisher« auf dem Amiga. Nach den Riesenerfolgen, die die Miniserie »The Punisher« von der Firma **Marvel Comics Group** bei weiten Teilen der Bevölkerung im englischsprachigen Raum hervorgerufen hat, und nach dem Kino-Tophit gleichen Namens, mit Dolph Lundgren als Hauptakteur, wurde nun das Spiel »The Punisher« auf dem Amiga kreiert.



Optisch ansprechend – »Ports of Call«

Neue Scanner-Produkte

Die von ASDG entwickelten Scanner-Pakete, Scanlab 100 und Professional Scanlab, erlauben das Einlesen von Farbbildern in höchsten Auflösungen auf allen Amiga-Modellen. Die Software unterstützt über 16 Millionen Farben, wobei die Berechnung der Bilder von einer festgelegten Farbpalette ausgeht.

Eine weitere Möglichkeit ist das Einlesen aller gängigen Grafikformte, wie Bilder von Digi-View, dem Framebuffer von Mimetic, Raytracing-Bilder von Sculpt 4D, Turbo-Silver, ARES und Dynamic High-Res, Farbseparation für 12- und 24-Bit-Farbbilder. Damit die Datenaufnahme keine Grenzen kennt, wird eine 80-Megabyte-Festplatte und 8-Megabyte RAM-Erweiterung empfohlen.

Dabei ist das ganze kompatibel zu Video-Transportcontrollern, die eingescannte Bilder direkt an den Videorecorder weitergeben, um beste Qualität bezüglich Auflösung und Farbe zu erhalten.

Info: CompuStore GmbH
Fritz-Reuter-Str. 6
6000 Franfurt 1

Neue Software-Kataloge von Commodore

Seit kurzem sind zwei neue Software-Kataloge von Commodore erhältlich: einen für den MS-DOS-Bereich und einen für den Amiga.

Im Amiga-Katalog findet der interessierte Benutzer alles über neues Zubehör, wie Programme oder andere Accessoires. Über wenig Vielfalt braucht sich niemand zu beschweren, da über 900 Hard- und Software-Produkte für alle denkbaren Anwendungen angeboten werden.

Preis: 19,90 DM

Info: Fachhandel oder
Markdienste Haberbeck
Postfach 1420
4937 Lage/Lippe

Prozessorkarte 68030 für Amiga 2000

Um dem Amiga 2000 auf die Sprünge zu helfen, bietet Commodore eine Turbokarte (A2630) auch einzeln an.

Die Karte bietet außer einer 25 MHZ getakteten Motorola 68030 den dazugehörigen arithmetischen Koprozessor 68882 sowie 2 Megabyte 32-Bit RAM, die nachträglich auf 4 Megabyte ausgebaut werden können. Die Karte wird einfach in den freien CPU-Slot des Amiga gesteckt.

Infos: Commodore Büromaschinen GmbH, Lyoner Str. 38, 6000 Frankfurt 71

Neuer Datenbank-Manager

Um in unserer modernen Zeit Daten jeglicher Art zu erfassen, wird ein entsprechendes Programm benötigt.

Die heutigen Programme können aber mehr als nur Adressen von Bekannten oder Freunden verwalten. Ein neuer Datenbank-Manager erfaßt nicht nur persönliche Daten, sondern kann Grafiken oder gesampelte Musik in einer einzigen Datei ablegen. Weitere Features sind: eine Finanzverwaltung, vorgefertigte Schablonen für diverse Anwendungen, Ausdrucke von vom Benutzer vordefinierten Etiketten, Anzeigen und Speichern von IFF-Files jeder Art, egal ob es sich um Grafiken von D-Paint oder Digitalisierprogrammen handelt, Einbinden von digitalisiertem Sound vom IFF-Format, wie Future-Sound, Pro Sound Designer oder kompatiblen Programmen. Damit steht einer eigenen Dia-Show mit Soundunterlegung nichts mehr im Wege.

Preis: 84,– DM

Infos: Fachhandel oder
Rushware, Leisuresoft
und Profisoft

Jürgen Seibel

Nützliches für den Amiga

Disk-Utilities Nr.10

Hilfreiche Geister sind bei der Arbeit mit dem Amiga manchmal doch sehr gefragt, und billig sollen sie auch sein.

Daß der Amiga sich mittlerweile in allen Bereichen der Computeranwendungen etabliert hat, ist nicht von der Hand zu weisen. Jedoch hilft dieses Lob nicht über so manches Manko hinweg. Hiermit ist vor allem das AmigaDos angesprochen. Damit der Amiga nicht an Benutzerfreundlichkeit verliert, helfen nützliche Utilites. Nun liegt es an dem Benutzer, die für ihn relevanten Utilities aus dem großen Spektrum der Werkzeugkiste zu finden.

Aus der Reihe der Disk-Utilities von Markt & Technik wartet das Software-Paket Nr. 10 mit drei Utilites auf. Da wären DIMO – ein Diskettenmonitor, Recover II – zum Retten gelöschter Dateien und TUC – The Ultimate Cruncher.

Gehen wir zuerst auf **DIMO** ein. Dieses Programm ist ein Diskettenmonitor, der das Einlesen der gesamten Diskette erlaubt. Die eingelesenen Daten werden auf einem Editorfeld dargestellt. Nun können Sie die Tracks und Blöcke der Diskette einzeln auslesen und modifizieren. DIMO kann aus dem CLI und der Workbench gestartet werden. Der Bildschirm teilt sich in drei Bereiche. Im oberen Bereich findet sich ein Block mit drei Gadgets »QUIT«, »INFO« und »AUTO«.

Mit »QUIT« wird das Programm lediglich beendet. Dagegen ist die Option »INFO« schon etwas weitgreifender. Hierzu öffnet sich am unteren Bildschirm ein neues Fenster mit sage und schreibe dreißig Gadgets. Diese Gadgets kennzeichnen die im Editorfeld befindlichen Langworte. So erkennt der Benutzer, an welcher Stelle sich zum Beispiel die Checksumme, der Diskname oder die Extension befindet. Das Gadget »AUTO« birgt nur zwei Möglichkeiten. Entweder wird das Auto-Read ein- oder ausgeschaltet. Bei einer Änderung des Sektors von Heads oder Tracks liest das Programm automatisch die neuen Daten ein. Im mittleren Bereich präsentiert sich dem Benutzer das Editorfeld, welches die eingelesenen und modifizierten Daten auf dem Bildschirm zeigt. Die Optionen zur Steuerung des Programms befinden sich im untersten Teil. Hier zeigt das Programm die angeschlossenen Laufwerke an. Darunter befinden sich natürlich, falls installiert, auch die Harddisk und die RAM-Disk. Über die Maussteuerung können Sie nun die einzelnen Tracks, Sektoren und Blöcke ansprechen. Da die Disketten zweiseitig formatiert sind, muß noch der Head, entweder mit null oder mit eins, definiert werden.

Nachdem Sie die Read-Option angeklickt haben, liest das Programm die entsprechende Stelle der Diskette ein. Mit einer Mode-Funktion gelangen sie in den Eingabemodus. Hier können Sie die neuen Daten im ASCII-Format oder per Hextype eingeben und abspeichern. In einem Feld namens Driver zeigt das Programm an, welche Treiberroutine vom aktiven Device benutzt wird. Weitere Informationen bekommen wir durch das Typ-Feld. Hier wird angezeigt, welcher Blocktyp sich gegenwärtig im Editor-Buffer befindet.

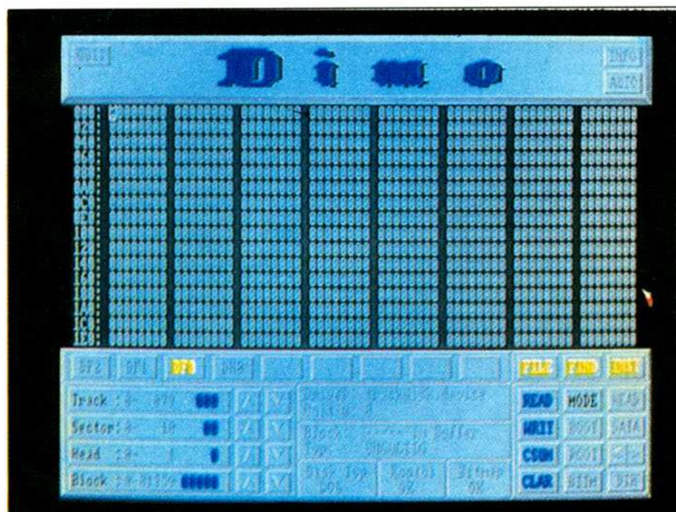
Nicht allein das direkte Ansprechen von Diskettenbereichen durch Block, Sektor und Trackdefinition ist möglich, sondern auch das Einlesen von Files. Dabei öffnet sich ein File-Requester mit der Anzeige des Directorys. Das Programm lokalisiert nun das angesprochene File auf der Diskette, so daß Änderungen sofort vorgenommen werden können. Nach der Modifizierung der Daten im Editorfeld berechnet der Computer die Checksumme und setzt sie an die erforderliche Position. Mit der Find-Option können Sie die Diskette nach bestimmten Begriffen durchforsten.

Kommen wir zum nächsten Programm. Ein überarbeiteter Amiganer kennt dieses Problem: Da sitzt man nun stundenlang vor seinem Amiga, um einen Quellcode zu schreiben. Nach dem Abspeichern will man sich seine Komposition noch einmal genüßlich ansehen. Doch was ist passiert? Statt der Load-Option hat man die Delete-Option angeklickt. Das Programm ist "futsch" – was nun? Hier soll **Recover II** der Retter in der Not sein. Das Programm ermöglicht das Wiederholen von gelöschten Daten. Dabei wird an der Diskette nichts geändert, wie es teilweise der Diskdoktor per Formatieren praktiziert. Recover II bietet vier Optionen: »Display«, »Save«, »Wipe« und »Break«. Die Option »Display« öffnet ein Fenster in dem die gelöschten Daten dargestellt werden. Mit der Option »Save« werden die gelöschten Daten in der RAM-Disk, auf der Diskette selbst oder gar auf einer dritten Diskette gerettet. »Wipe« löscht alle überflüssigen Datenreste auf der Diskette. Mit der Option »Break« können Sie die anderen Funktionen des Programms unterbrechen.

Das Programm **TUC – The Ultimate Cruncher** ist ein Packer. Aus einem File-Requester wählt man vorerst das zu packende File aus. Mit der Order »Pack es« beginnt das Programm mit dem Crunch-Vorgang. Zuletzt wird das gepackte File abgespeichert. Des weiteren können mit TUC gepackte Programme wieder entpackt werden, so daß man den ursprünglichen Quellcode erhält. Doch nun zu den Voreinstellungen, also den Flags, unter denen der Packvorgang abläuft. Da kann vorerst das File als Programm gepackt werden. Das bedeutet, daß das gepackte File aus dem CLI oder der Workbench gestartet werden kann. Dabei kann das gepackte Programm zusätzlich in das Chip-Memory gelegt werden. Die Option »Leise« schaltet das Flackern beim Entpacken ab.

Am unteren Bildschirm wird auf Wunsch die Effizienz des Crunchens errechnet. Zusätzlich wird dem Benutzer die Entpackadresse und die Startadresse gegeben. Eine Ergänzung zu TUC stellt das **TUC NewDOS** dar. Hierbei handelt es sich um ein Programm, das das AmigaDOS derart modifiziert, daß es in der Lage ist, gepackte Files während des Ladevorgangs zu entpacken. In dem Software-Paket präsentieren sich Utilities, die im Umgang mit Daten sehr nützlich sein können und unbedingt zum Handwerkszeug zu zählen sind.

(vb)



DIMO, der Diskettenmonitor – ein hilfreiches Utility

AMIGA DOS Blitzlicht

Name: Disk-Utilities Nr. 10
Vertrieb: Markt & Technik
Preis: 49,– DM

Positiv:
- benutzerfreundliche Menüs
- Erkennung der Langworte bei dem Programm DIMO
- hohe Crunch-Rate
- Zurückgewinnung von gelöschten Daten

Negativ:
- keine negativen Punkte anzumerken

Andreas Polk

Menübaustelle

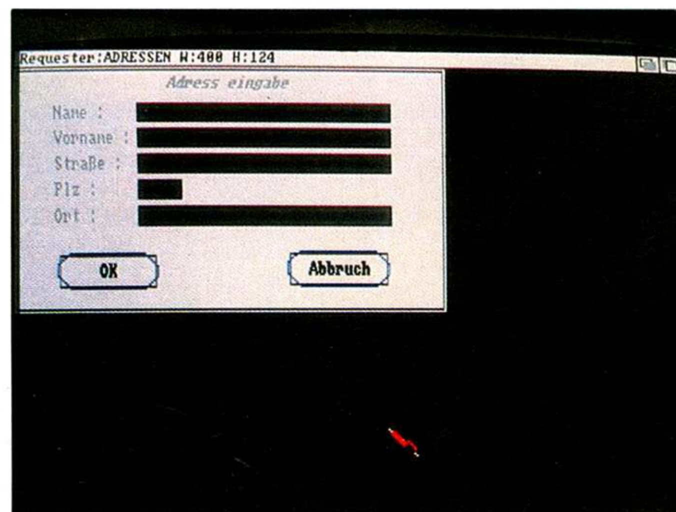
Das Requester/Menue Construction Tool

Wer den Amiga schon einmal unter Intuition programmiert hat, weiß, welche stupide Schreibarbeit dafür nötig ist. Wenn dann noch die richtigen Werte für Positionsangaben zu finden sind, vergeht viel Zeit. Es gibt allerdings gerade für solche Fälle eine Menge an Werkzeugprogrammen, so auch R.C.T.

R.C.T. ist eine Abkürzung für "Requester Construction Tool" und schon etwas länger auf dem Markt – trotzdem aber noch vielen nicht bekannt.
Wie der Name schon sagt, kann man mit dem R.C.T. also Requester und als Zugabe auch noch Menüs erstellen. Brauchen Sie ein Tool, mit dem Sie auch Windows oder Screens erstellen wollen, so müssen Sie auf andere Programme wie »Powerwindows 2.5« zurückgreifen; mit dem R.C.T ist dies nicht möglich.
Für 129,– DM erhält man ein deutsches Handbuch, das sich allerdings noch auf die Version 1.0 bezieht. Die Erweiterungen, die in der aktuellen Version 1.2 hinzugefügt wurden, müssen im Text "Neuerungen" nachgelesen werden, der sich auf der Diskette befindet. Das Handbuch umfaßt über 40 Seiten und gibt zu jedem Befehl kurze Erklärungen. Daß das Handbuch nicht jeden Schritt genau erklärt, ist sinnvoll, da sich ein Einsteiger kaum das R.C.T. zulegen wird. Allerdings hätten einige Dinge etwas genauer beschrieben werden können. Auf der Diskette befinden sich neben dem R.C.T. auch noch einige fertige Beispiele, das Programm, um den Sourcecode für verschiedene Programmiersprachen zu erstellen, sowie der Programmteil »R.C.T.Lab«.

Der Requester-Editor des Tools – Entscheiden Sie sich!

Das Programm unterteilt sich in zwei große Programmteile, den Requester-Editor und den Menü-Editor. Im Requester-Editor erscheint der Requester als weiße Fläche. Sie kann mit der Maus wie ein Fenster vergrößert und verkleinert werden. Dieses Feld ist das 'Spielfeld' für die nun folgenden Möglichkeiten. Der Requester-Editor erlaubt es dem Benutzer, alle möglichen Arten von Gadgets in diesem Requester unterzubringen. Per Mausklick läßt sich ein Boolean-, String- oder Proportional-Gadget erzeugen. Diese Gadgets sind dann auf vielerlei Art modifizierbar. Durch Anwahl entsprechender Menüpunkte lassen sich die Klickbereiche, die Gadget-Texte, die Farben und vieles mehr ändern. So bietet das R.C.T. beispielsweise bei der Textmanipulation die Möglichkeit, den Text in einem beliebigen Font darzustellen. Auch Textattribute wie fett, kursiv und so weiter sind ohne weiteres möglich. Hier sind dem Anwender kaum Grenzen gesetzt.
Eine Besonderheit ist sicherlich noch der Border-Editor. Haben Sie ein Gadget erstellt, so können Sie im Border-Editor den Border dieses Gadgets nach Belieben verändern. Der Editor bietet dazu einige Sonderfunktionen, wie beispielsweise die Symmetrie oder Hilfslinien. Er kann bis zu fünf Borders verwalten, die alle wahlweise aus- und eingeschaltet werden können. Hiermit werden dem Anwender vielfältige Möglichkeiten geboten. Ein weiteres Feature des R.C.T. ist die Möglichkeit, ein Gadget mit einem Muster oder einem IFF-Bild zu hinterlegen. Zuerst muß für das Gadget das entsprechende Flag gesetzt werden. Dies geschieht ganz einfach in einem dafür vorgesehenen Requester, in dem sich übrigens alle Flags setzen und löschen lassen.
Damit der Anwender auch weiß, was die einzelnen Flags bedeuten, ist jedes im Handbuch genau erklärt – sehr lobenswert! Anschließend lädt man eine IFF-Grafik ein und hinterlegt das Gadget mit einem Teil des Bildes. Auf diese Weise ist es auch möglich, daß ein Gadget beim Anklicken ein anderes Bild beziehungsweise Muster darstellt. Natürlich kann auch ein ganz normaler Intuition-Text in einen Requester eingegeben werden. Nach Aufruf einer entsprechenden Funktion erscheint dann der Standardtext, der genauso wie die Gadgets in vielfältiger Weise verändert werden kann. Haben Sie einen Requester erstellt, so können Sie ihn nach Anwahl eines Menüpunkts austesten. Er verhält sich nun so, wie er sich auch im Programm geben würde. Per Mausklick können dann wieder Veränderungen vorgenommen werden, falls einzelne Teile noch nicht so arbeiten, wie dies zu wünschen ist.
Der Menü-Editor ist der zweite Programmteil des R.C.T. Auch hier lassen sich die Menüs auf vielfältige Art erzeugen und verändern. So sind für die Menüs verschiedene Schriften wählbar. Auch die Klickbereiche sind frei erstellbar.
Untermenüs sind natürlich auch kein Problem. Der Menü-Editor bietet dem Anwender alles, was auch über die direkte Programmierung möglich ist.

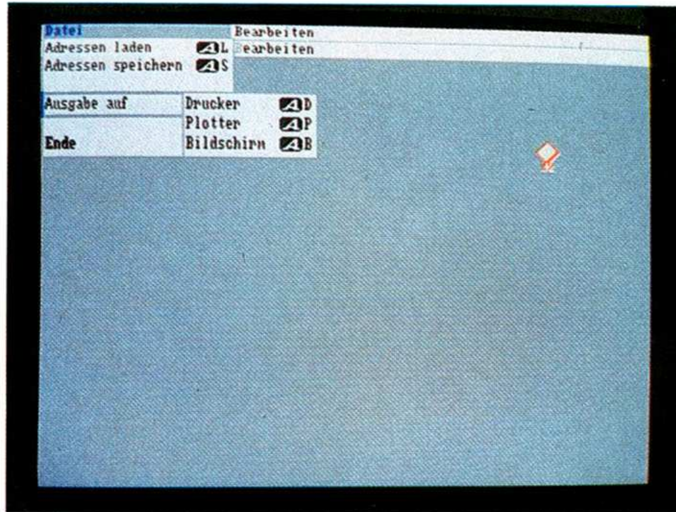


Der Requester-Editor des R.C.T.

Der Menü-Editor – Was hätten Sie denn gerne?

So lassen sich hier ebenfalls alle Flags frei einstellen. Lediglich das Flag »Mutual Exclude«, welches zum Deaktivieren anderer Menüpunkte bei Anwahl des aktuellen Menüpunkts gilt, wird im Handbuch nicht erläutert. Somit wird diese Funktion leider nicht nutzbar, denn woher weiß das R.C.T., welche anderen Menüpunkte deaktiviert werden? Wahrscheinlich muß der Anwender dies beim Programmieren selber noch abändern. Damit ein Menü nicht zu ungeordnet aussieht, lassen sich über den Menüpunkt »Justieren« alle Menüpunkte einstellen. Das ist sehr sinnvoll, da beim schnellen Arbeiten leicht einmal ein Menüpunkt verrutscht. Allerdings arbeitet diese Funktion etwas merkwürdig. Manchmal werden die Menüpunkte justiert und manchmal nicht.
Um eine erstellte Oberfläche in das eigene Programm einzubinden, werden Ihnen zwei Möglichkeiten geboten. Die erste besteht darin, mit Hilfe des Programmes »RCT_TO_ASCII« aus der R.C.T.-Datei den Source-Code für die Programmiersprachen BASIC, GFA-BASIC, C und Assembler zu erstellen.
Die Generierung geht schnell vonstatten. Die so entstandenen Strukturen müssen jetzt noch in das eigene Programm eingebunden werden, und entsprechende Funktionen, die Sie selber programmieren müssen, binden die Oberfläche dann ein. Die zweite Möglichkeit, die Oberfläche in das eigene Programm einzubinden, ist über die »rct.library« realisierbar. In dieser Library sind einige Funktionen enthalten, die Ihnen die Einbindung der Oberfläche auf einfache Weise abnehmen. Wird eine Oberfläche auf diese Art und Weise erstellt, so muß sich beim Start des Programms später immer die »rct.library« in dem Verzeichnis »libs:« befinden. Da diese Library ko-

piert werden darf, ist es auch nicht illegal, das selbsterstellte Programm mit der Library weiterzugeben. Neben einigen Demos befinden sich noch zwei weitere wichtige Dinge auf der Diskette, zum einen eine Symboldatei und ein Demoprogramm zum Einbinden der mit dem R.C.T. erstellten Oberfläche in Modula2. Dies ist bislang aber nur über die Funktionen der »rct.library« möglich. Hier wäre die direkte Source-Code-Generierung für M2-Amiga noch wünschenswert, denn immerhin führt diese Programmiersprache schon lange kein Mauerblümchendasein mehr. Zum anderen kann über das Programm »R.C.T.Lab« eine erstellte R.C.T.-Datei ausgetestet werden, ohne das R.C.T. selber starten zu müssen. Eine Veränderung der R.C.T.-Datei ist nur dann möglich, wenn sich eine bestimmte Datei auf der Diskette befindet. So kann man Oberflächen an andere Personen weitergeben, ohne daß sie diese verändern können, selbst wenn sie auch im Besitz des R.C.T. sind. Trotz der wirklich zahlreichen Funktionen bleibt ein gemischtes Gefühl zurück.



Der Menü-Editor des Requester Construction Tool

Was kann man abschließend über das Werkzeug sagen?

Zwar bietet das R.C.T. umfangreiche Möglichkeiten, allerdings hat man oft den Eindruck, daß die Programmierer zuviel des Guten vorhatten. Möchte man einfach mal schnell ein "normales" Menü ohne großen Schnickschnack erstellen, so bietet das R.C.T. hier zuviel an Einstellungen. Die Justierung klappt nicht immer so, wie man es gerne hätte, und meiner (subjektiven) Meinung nach werden die einzelnen Menüpunkte auch zu dicht untereinander dargestellt. Dies wäre jedoch nicht so schlimm, wenn sich das R.C.T. nicht so oft mit einem Guru verabschiedet hätte. Das darf in einem solchen Programm nicht passieren.

(jb)

AMIGA DOS Blitzlicht

Name: Requester Construction Tool
Preis: 129,– DM
Anbieter: Maxon Computer GmbH, Industriestr. 26, 6236 Eschborn, Tel.: 06196/481811

Positiv:
- viele Gestaltungsmöglichkeiten
- schnelle Source-Code-Generierung
- viele Beispiel- und Zusatzprogramme im Lieferumfang

Negativ:
- stürzt zu häufig ab
- erzeugt keine Fenster und Screens
- keine Source-Code-Generierung für Modula2



Im Quartett klingt's nett!

Soundeditor »Quartet« – Musik für jedermann

Seit die AMIGA DOS besteht, sind im Heft Tests über Sound-Editoren erschienen. Viele Firmen haben ein Herz für Musiker und solche, die es gerne wären, entdeckt, und so haben gerade die Programme, die Musik auf einfachste Weise erklingen lassen, Hochkonjunktur. In diese Gruppe reiht sich auch ein neues Produkt der Firma Microdeal ein, das wir Ihnen hier vorstellen wollen: Quartet – The digital music machine.

Seine musikalischen Fähigkeiten sind einer der Hauptgründe, warum der Amiga ein Erfolg auf dem Computersektor wurde. Hauptsächlich spielen dabei die Möglichkeiten der Digitalisierung eine Rolle, der Soundchip PAULA ist in dieser Beziehung fast unschlagbar. Die Möglichkeit, digitalisierte Sounds wiederzugeben, ist denn auch ein wesentlicher Faktor, warum Sound-Editoren mit gesampelten Instrumenten, also Instrumenten, die digitalisiert wurden, auf dem Amiga weit verbreitet sind. In diese Kategorie reiht sich auch Quartet ein.

Wie wär's mit dem »Mozart.Set«

Quartet unterscheidet sich in einigen Punkten von herkömmlichen Soundeditoren. Da wäre erst einmal das Disketten-Menü zu nennen. Während bei bekannten Soundprogrammen wie dem Soundtracker oder Sidmon die einzelnen Instrumente nacheinander geladen werden müssen, kann man hier die Instrumente zu »Sets« zusammenfassen und auch als Gruppe abspeichern. Der Vorteil: Während die anderen Programme auf einen festgelegten Instrumentenpark zurückgreifen und einzelne Samples sich nur durch Dazuladen ändern lassen, können hier für ein und dasselbe Musikstück unterschiedliche Sets von Instrumenten geladen werden, ohne daß noch viel zu ändern wäre. Außer diesen Sets lassen sich natürlich auch einzelne Samples, die sich auf einer mitgelieferten Diskette befinden, laden und nutzen.

Die Musikinstrumente belegen die Zehnertastatur, das heißt, jedem Instrument ist eine Taste des Zehnerblocks zugeordnet, außer den Tasten [0] (Ins) und [.] (Del). Einzelne Instrumente lassen sich daher auf die entsprechende Taste bringen. Damit ist aber auch klar, daß nur 16 Instrumente benutzt werden können, wo-



Bild 1. Etwas ungewöhnlich, aber es funktioniert: Die Lade- und Speicheroperationen werden in einem separaten Screen erledigt – Copper-Spiele gibt es gratis dazu

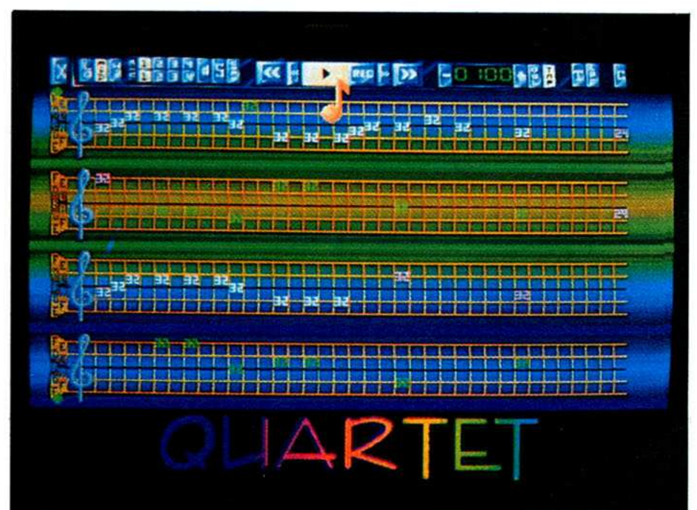


Bild 2. Der Editor mit den vier Kanälen und den enthaltenen Instrumenten. Diese bestehen aus Zahlen und werden notenähnlich eingesetzt

bei das »nur« wohl ausreichen dürfte. Diese Instrumente bekommen die Nummern 11 bis 44 zugewiesen.
In diesem Disketten-Menü lassen sich also Sets und Samples laden und sichern. Natürlich gilt das gleiche für Musikstücke, auch sie können gespeichert und wieder in den Editor geladen werden.
Samples können direkt vom Sampler übernommen werden; der in Großbritannien schon bekannte A.M.A.S.-Sampler, der mittlerweile auch in Deutschland vertrieben wird, kann diese gesampelten Sounds liefern. Aber auch andere Marken-Sampler können benutzt werden.
Die Instrumentensets zu erstellen ist relativ einfach. Die Sets sollte man sich vorher abspeichern, um im Bedarfsfall sofort darauf zugreifen zu können. Außerdem kann man sich so seine Lieblingsinstrumente zusammentragen.
Der Editor von Quartet ist ebenfalls etwas gewöhnungsbedürftig. Während man von anderen Soundeditoren das Arbeiten mit Patterns und Tracks, also die Aufsplittung des Songs in einzelne Teile, gewöhnt war, findet man im Quartet ein Notenblatt, auf dem sich vier Eingabefelder befinden, die jedes für sich einen der vom Amiga nutzbaren Kanäle darstellt. Die Noteneingabe geschieht nun so, daß man sich eine bestimmte Instrumenten-Nummer aussucht, entweder wird mit der Maus die Nummer angeklickt oder die entsprechende Taste auf dem Zehnerfeld gedrückt, und dieses Instrument als Note einsetzt. Dabei sind auch Halbtonschritte möglich. Alle Funktionen des Programms befinden sich in einer Menüleiste, die sich jedoch bildlich darstellt und die Funktionen per Knopfdruck parat hält. Die Noten lassen sich in drei Oktaven eingeben, dazu dienen die Buttons »Lo« (Oktave 1), »Mid« (Oktave 2) und »Hi« (Oktave 3). Außer den Instrumenten-Nummern befinden sich noch Buttons für Halbtonschritte, für die Lautstärke des Instruments, für die Geschwindigkeit des Musikstücks und das Anheben oder Absenken des kompletten Musikstücks und der obligatorische Ein-/Aus-Schalter für den Filter, der optisch auch durch die Power-LED repräsentiert wird. Das editierte Musikstück kann auf vielerlei Art verändert und verbessert werden; eine Slide-Funktion erreicht zum Beispiel das Auf- und Abschwellen zum nächsten Ton. Über die [Space]-Taste läßt sich der polyphone Betrieb einschalten. Hier kann über ein Keyboard, das über ein MIDI-Interface mit dem Amiga gekoppelt wird, der Song begleitet werden. Falls kein Keyboard vorhanden ist, kann man die eigentliche Amiga-Tastatur als Keyboard benutzen.
Bei der Vielzahl von Soundeditoren, die es inzwischen gibt, muß man Quartet einen Sonderplatz einräumen und dies vornehmlich durch die MIDI- und Sampler-Optionen. Der Editor ist allerdings ein bißchen umständlich zu bedienen, zwar können die Sounds ebenfalls in Patterns aufgeteilt werden, der gesamte Sound ist jedoch immer in einem Stück zu sehen. Durch das Tastenfeld mit den recorderähnlichen Funktionen bekommt man allerdings schnell Eingang zum Programm, Funktionen wie »RECORDING« und Kopieren helfen beim Einstieg schon eine ganze Menge. Einzig die Sample-Diskette gibt noch Anlaß zur Kritik: Zwar sind die meisten gesampelten Instrumente recht vernünftig, was jedoch die willkürlich aus richtigen Musikstücken herausgeschnipselten Teile auf der Diskette sollen, weiß wahrscheinlich nicht mal Mozart. Hier wird man dazu übergehen, sich eigene Instrumente zu sampeln oder auf fertige Disketten zurückzugreifen.
Ansonsten ist das Programm ganz ordentlich und verdient die Note gut.

(jb)

AMIGA DOS Blitzlicht

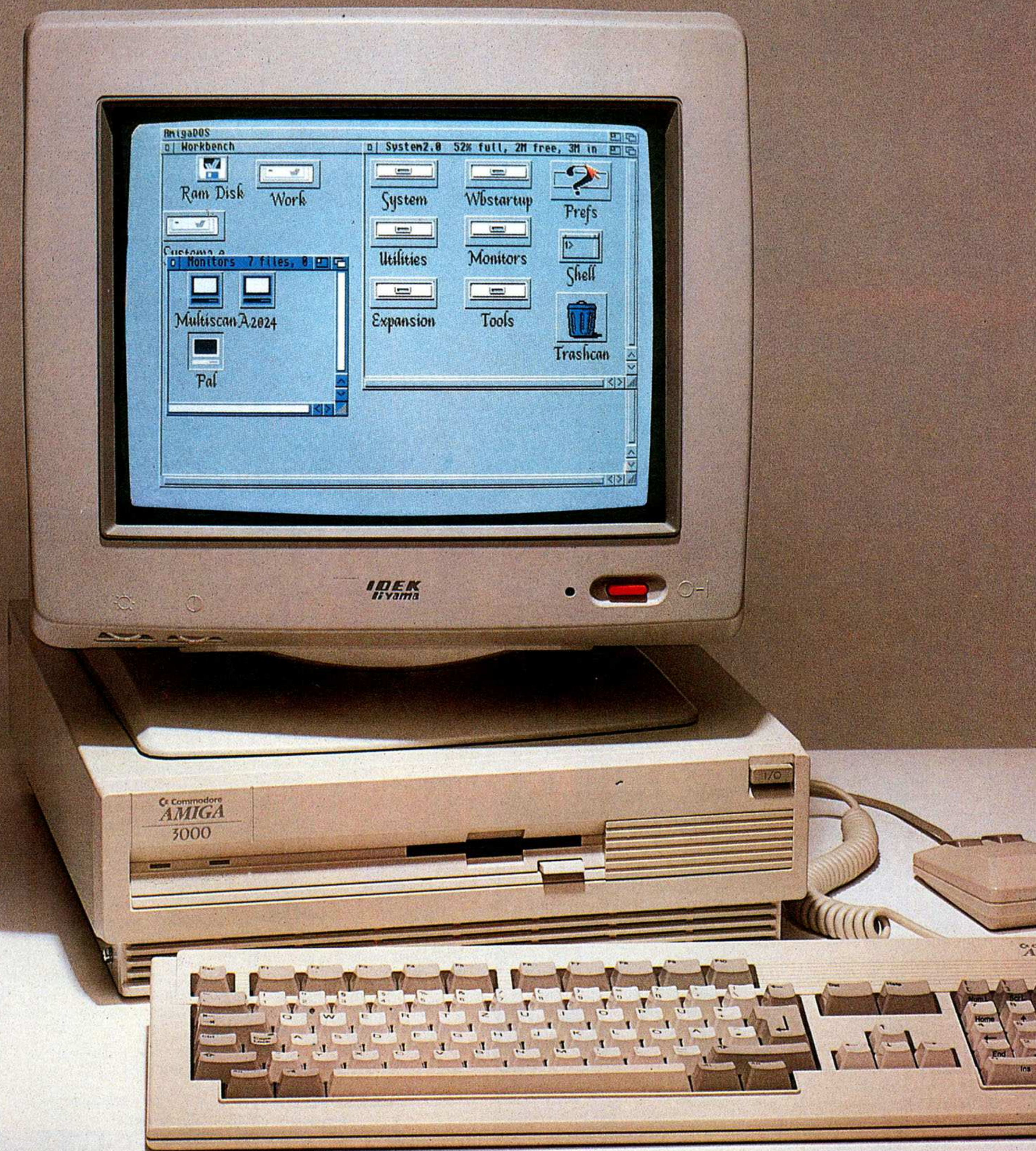
Name: Quartet
Vertrieb: Microdeal
Funktion: Soundeditor
Preis: stand bei Redaktionsschluß noch nicht fest.

Positiv:
- MIDI-fähig
- Samples können sofort über externen Sampler eingelesen werden
- polyphones Spielen möglich
- leichte Noteneditierung

Negativ:
- zu unflexibel in der Handhabung
- zuwenig Samples auf der Diskette, dafür zuviel unsinnige Mitschnitte

Begegnung mit einem Traumcomputer

Der Amiga 3000 im ersten Test



Erstaunliches hatte man von ihr gehört: Sie sollte viel besser aussehen als ihre Vorgängerinnen, ihre Arbeit schneller verrichten als je eine andere ihrer Familie. Sie sollte alles mitbekommen, was jemanden wie sie begehrenswert macht. Wovon hier die Rede ist? Nein, nicht von einer neuen Redaktionssekretärin, sondern von einer neuen Freundin – Modell 3000.

Das Auspacken eines Paketes ist fast immer eine spannende Sache, bei den beiden neu angekommenen Paketen wurde diese Prozedur fast zu einer Feierlichkeit. Der Supercomputer der neunziger Jahre war angekündigt worden und pünktlich eingetroffen. Gehört hatte man viel, gesehen dafür weniger, umso größer waren die Erwartungen. Und so stand die gesamte Redaktion der AMIGA DOS um den Platz, der für den "Traumcomputer" unter den Amiga-Modellen gedacht war.

Neue Features: Super-Prozessor und liegende Erweiterungskarten

Gut ausgestattet ist er, der A3000. Ein 68030 verrichtet hier seine Arbeit im Zusammenspiel mit einem 68881- oder 68882-Coprozessor. Getaktet werden beide mit 25 oder 16 MHz, damit ist Schnelligkeit Trumpf. Die Grundversion verfügt über 1MByte Chip-RAM und 2 MByte Fast-RAM, kann jedoch aufgerüstet werden auf 2 MByte Chip-RAM und 16 MByte Fast-RAM. An Speicher mangelt es dem 3000er also nicht.

Die restliche Peripherie ist allerdings auch nicht ohne. Ein Super-Fat AGNUS ist für die exzellente Grafik zuständig; unterstützt wird er von 1 MByte Chip-RAM. Auch sonst hat sich einiges auf der Platine getan; mehr hochintegrierte Bausteine sind zu finden, weniger diskrete Bauteile, wie Transistoren und Dioden. In Punkto Videoausgänge findet man einen neuen Port: Die Video-Signale werden nicht nur auf den RGB-Port für den "normalen" Amiga herausgefüht, zusätzlich besitzt der A3000 von Hause aus noch eine fünfzehnpolige Sub-D-Buchse mit zum VGA-Port des PC kompatiblen RGB-Signalen. Hier können zum Beispiel Multiscan-Monitore angeschlossen werden. Zu den Monitoren, oder besser zur Auflösung, kommen wir später noch.

Auch ein SCSI-Port für externe Festplatten oder Tape-Streamer ist zu finden. Hier können alle Peripherie-Geräte angeschlossen werden, die über einen SCSI-Port verfügen (Small Computer Standard Interface – findet heute vor allem bei Massenspeichern, wie Fest- oder Wechselplatten, und auch Bandlaufwerken, wie Tape-Streamern,Verwendung).

Das technische Design des A3000 erinnert mehr denn je an ein professionelles Gerät; von der verspielten Einfachheit eines A500 ist hier nichts, aber auch gar nichts zu merken, und selbst der A2500/30 verblaßt angesichts dieser geballten Ladung von High-Tech-Komponenten.

Auch das Aussehen des A3000 unterscheidet sich grundlegend von dem seiner Vorgänger A2000/A2500. Nicht mehr klobige Blecharchitektur ist angesagt, sondern frisch vom Designerstudio importierte Gehäuse-Hardware. Im Ernst, der Unterschied zwischen Amiga-Oldtimer und Amiga-Newcomer ist schon immens; der A3000 fällt durch fast klein zu nennende Abmaße auf. Und hier ist gleich noch ein Novum: Die Erweiterungskarten werden jetzt (notgedrungen) quer eingebaut. Insgesamt sind es vier Steckplätze, über die der Besitzer für diverse Erweiterungen verfügen kann. Sie stehen allerdings dicht beieinander, so daß beim Einsatz der Karten mit einem plötzlichen "Steckplatz-Schwund" zu rechnen ist, wenn die Bauhöhe der Karten zu groß ist.

Unser Testgerät verfügt von Hause aus über eine 40-MByte-Festplatte und ein internes 3,5-Zoll-Laufwerk. Ein zweites Laufwerk, ebenfalls in der 3,5-Zoll-Baugröße, kann als »Df1:« eingebaut werden, der Platz dafür ist vorhanden. Wie schon bei den Vorgängermodellen findet sich auch hier an der Rückseite eine Steckbuchse für zusätzliche Laufwerke in den Baugrößen 3,5 oder 5,25 Zoll. Das hier angeschlossene Laufwerk bekommt automatisch die Kennung »Df2:«, da das zusätzliche interne Laufwerk fest auf die Gerätekennung »Df1:« eingestellt ist. Weitere Ports sind wieder einer für parallele Übertragung, einer für serielle und ein Videoport für Amiga-Monitore. Tastatur, Maus und Joystick werden seitlich mit dem Rechner verbunden, vorne befinden sich nur die Laufwerksschächte und (ein Grund zum Jubeln) der Netzschalter. Das umständliche "Hinter-den-Rechner-Greifen" gehört damit der Vergangenheit an.

Die Tastatur besitzt zwar ebenfalls ein neues Gehäuse, von der Gestaltung her der Zentraleinheit angepaßt, sie selbst hat sich jedoch nicht verändert. Die Anordnung der Tasten ist gleich geblieben. Bei der Maus hat sich ebenfalls nichts geändert, hier hätte man sich allerdings eine Neuerung gewünscht.

Der A3000 braucht eine Kickstart-Diskette

Das Einschalten des A3000 beginnt mit einer Überraschung. Die vorinstallierte Festplatte meldet das Booten einer Kickstart. Wo ist das ROM? Bei genauerer Betrachtung fällt es auf: Das bisher bekannte Kick-ROM fehlt! Und ein Blick auf die mitgelieferte Software bestätigt den ersten Eindruck; eine Kickstart 2.0 liegt dem Paket bei. Hier scheint man die gute Idee, die man beim Amiga 1000 hatte, nämlich variabel bei den Betriebssystemen zu sein, wieder aufgegriffen zu haben. Das Kickstart-System wird von der Platte in einem festgelegten RAM-Bereich geladen und dort initialisiert.

Das ist jedoch nicht die einzige Überraschung, die der A3000 von sich gibt. Kaum ist die Workbench geladen, traut man seinen Augen kaum. Ein völlig neues Aussehen überrascht den Amiganer. Screens, die in ihrem Aufbau und ihren Farben an einen dreidimensionalen Untergrund denken lassen, Windows, die in einer Statuszeile Blockbelegung, Anzahl der Directories und Dateien anzeigen, Hauptverzeichnisse, die den freien Speicherplatz der Diskette oder Festplatte mitteilen, überarbeitete Gadgets, auf der Workbench änderbare Fonts, Grafikmusterunterlegung – man ist erst einmal förmlich erschlagen über die Unterschiede zum "alten" System. Doch der Reihe nach!

Nach dem Laden der Workbench meldet sie sich im Interlace-Modus, dem der normale 1084er Monitor nicht gewachsen ist. Um das Geflimmer wegzubekommen, schaut man deshalb zuallererst im Verzeichnis »Prefs« nach »Preferences« aus, um eine andere Betriebsart einzustellen. Hier erwartet einen die nächste Überraschung: »Preferences« gibt's nicht mehr. Statt dessen erwarten einen 12 Icons, wovon jedes ein Programm zur Einstellung irgendeines Systemteils darstellt.

Das Grafikwunder – mögen Sie es unterlegt?

»Input« dient zur Einstellung der Tastatur und Mausgeschwindigkeit, »Printer« nimmt sich der Druckertreiber an, »Serial« sorgt für reibungslose Übertragung über die serielle Schnittstelle und weiteres mehr. Hier ist noch alles wie gehabt, jedoch »WBscreen« erzeugt schon wieder Überraschungsrufe. Statt nur »Interlace« oder »Hires« anzuzeigen, wie man das gewohnt ist, findet man hier noch »SuperHires« und »SuperHires-Interlaced«, wobei man es nicht lassen kann, statt »Hires« (augenschonend) den "supergehiresintergelacten" Bildschirm einzustellen. Das Ergebnis? 1280x512 Bildschirmpunkte und eine fast akute Bindehautentzündung dank des nicht entspiegelten 1084er Monitors. Hier wird es deutlich, wer A(miga) sagt (und eine hohe Auflösung haben möchte), muß jetzt doch auch M(onitor) sagen. Um eine Neuanschaffung kommt man bei höheren Ansprüchen nicht herum. Für die diversen Monitore, seien es Multiscan-Ausführungen oder der neue A2024-Monitor, sind diverse Ansteuerprogramme vorhanden, in denen der genutzte Bildschirm angemeldet werden kann.

Der 1084 verrichtet im Hires-Modus allerdings auch seine Dienste, zeigt jedoch auch schnell seine Grenzen. An eine Arbeit im »SuperHires-Interlaced«-Modus ist nicht zu denken, und wenn, dann nur über Verkleinerung des sichtbaren Bereichs. Ja, auch das ist möglich, der Workbench-Bereich kann eingegrenzt werden. Geht er über den sichtbaren Bereich hinaus, gibt es eine Autoscroll-Funktion, die in die "unerforschten" Bereiche vorstößt. Und um zu unserer Frage aus der Zwischenüberschrift zurückzukommen, das mit dem Unterlegen ist kein Scherz. "WBPattern" sorgt für grafische Untermalung des Screens oder der Windows. Entweder man nimmt vorgefertigte Muster, oder man läßt seine Fantasie spielen; 16 Farben stehen für die Muster im Hires-Modus parat, vier im SuperHires-Interlaced.

Diese ganzen grafischen Einstellungsprogramme dienen eigentlich nur dazu, den Amiga-Screen zu etwas Unnachahmlichen zu machen, aber genau das ist garantiert auch der Grund, warum man darauf zugreift. Die Amiganer mögen's ja schon immer bunter und lauter als die anderen Computerbesitzer, gelle?

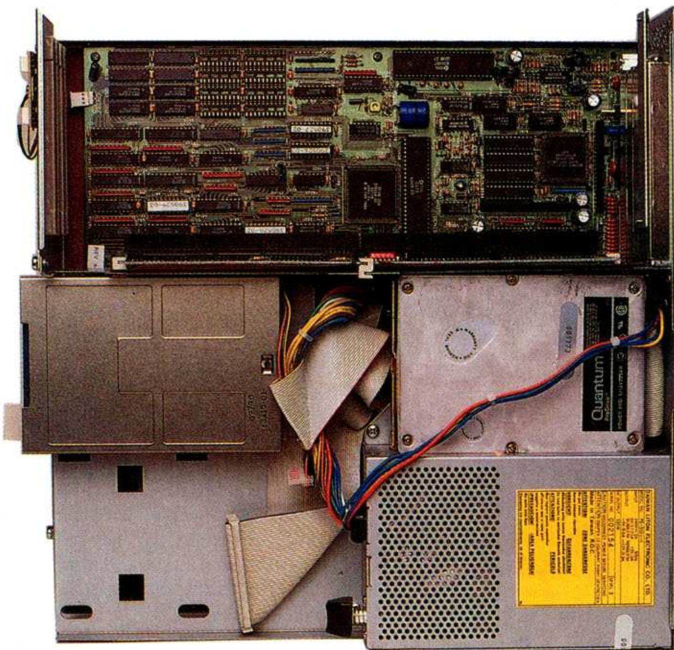


Bild 1. Das Innenleben des A3000. Steckkarten finden nur noch liegend eingebaut Platz

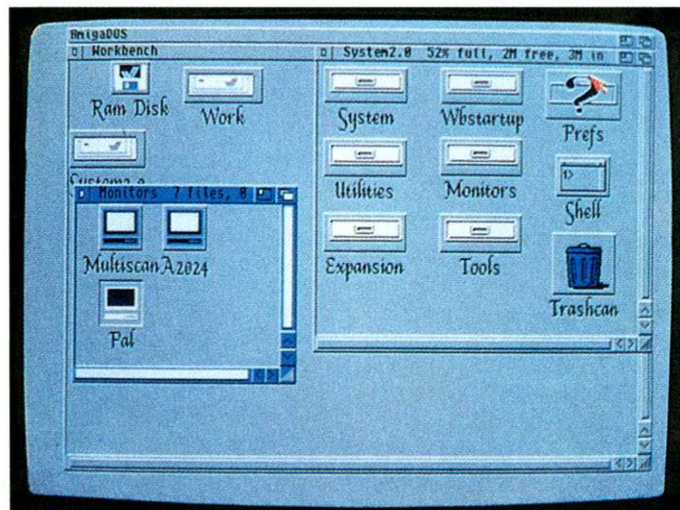


Bild 2. Die neue Workbench 2.0. Nicht nur Zeichensätze lassen sich auswählen...

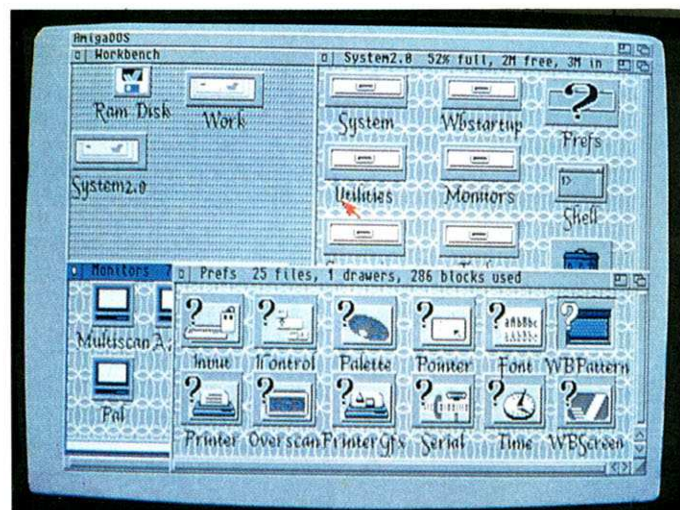


Bild 3. ...sondern auch Hintergrundmotive für Screen und Windows

Verbesserungen, Neues und Altbewährtes

Nicht nur grafisch hat die neue Workbench einiges zu bieten, auch bisher etwas "unglückliche" Programme wurden überarbeitet. Der Icon-Editor zum Beispiel läßt sich diesmal direkt sagen, welche Icon-Art erstellt werden soll. Außerdem erlaubt er jetzt den Einsatz von IFF-Brushes. Man sieht förmlich die kunstvoll gestalteten Piktogramme vor sich, wenn man an die Möglichkeiten denkt, die sich da auftun.

Der CLI befindet sich zwar immer noch im System-Directory, beim Öffnen befindet man sich jedoch wieder in der Shell. Shell und CLI sind hier ein und dasselbe. In beiden können nun die Vorzüge genutzt werden, die in der Workbench 1.3 nur der Shell zugedacht waren.

Die Startup-Sequence ist ebenfalls runderneuert worden. Zwar existieren immer noch die Patch-Files »dpat« und »spat«, die »Startup-II« ist jedoch nicht mehr zu finden; eine Aufteilung findet damit nicht mehr statt, sie ist mit der neuen Kickstart-Version auch hinfällig geworden.

Das Shell- oder CLI-Fenster kann jetzt auch per Maustaste geschlossen werden, wer es lieber ohne Shell mag, aber auf die DOS-Befehle nicht verzichten will, kann diese von der Workbench aus eingeben. Im Workbench-Menü hat sich auch einiges geändert, die Menüs sind "reichhaltiger" geworden, das heißt, wesentlich mehr Möglichkeiten stehen

hier zum Umgang mit Piktogrammen und Fenstern zur Verfügung.

Der Traumcomputer – real besehen

Er ist eine Überraschung – das muß man wohl angesichts des neuen Aufbaus und der völlig überarbeiteten Version der Workbench sagen. Ein paar Wermutstropfen traten allerdings auch zutage: Nicht jedes Programm aus unserem Software-Pool arbeitete reibungslos mit dem A3000 zusammmen und weigerte sich beharrlich zu laufen. Dabei handelte es sich um bekannte Grafikprogramme (Ausnahme »Deluxe Paint III«, das mit einer unglaublichen Geschwindigkeit Flächen mehrfarbig füllte), Anwendungen und auch Spiele, von denen die Mehrzahl Probleme hatte. Ob es am Prozessor oder an der neuen Kickstart liegt? Tatsache ist, daß an der Software zum A3000 noch gearbeitet wird, dies also kein Dauerzustand sein dürfte. Ansonsten macht dieser “Super-Amiga“ richtig Spaß; irgendwie kam wieder das Amiga-Feeling durch, das man damals bei den ersten Tausendern hatte.

Amiga 3000 oder 68030-Karte für den A2000?

Genug der Schwärmerei, versuchen wir jetzt einmal diese Neuerscheinung auf dem Computermarkt unter realen Gesichtspunkten zu sehen. Ab Anfang Sommer soll das neue Flaggschiff der Amiga-Reihe in den Handel kommen. Die Preise sollen dabei zwischen 5000 und 8000 DM liegen, je nach Ausführung des A3000 (68030 + 68881 mit 16 MHz Taktfrequenz und 40 MByte Festplatte oder 68030 + 68882 mit 25 MHz Taktfrequenz und 105 MByte Festplatte). Ersatzweise steht der »Pseudo-A3000« dagegen, ein Amiga 2000 mit 68030-Karte, eingebauter Festplatte und mindestens drei MByte RAM. Rechnet man beides zusammen, kommt man beim Amiga 3000 besser weg, allerdings je nach Angebotslage. Wie sieht es mit der »Gebrauchsfähigkeit» aus? Ein A2000 mit Prozessorkarte hat immer noch den Vorteil, einen 68000 als Grundprozessor zu besitzen, Programme laufen also soweit uneingeschränkt. Der Amiga 3000 besitzt ein Kickstart-Menü und zwei Kickstart-Partitionen auf der Festplatte. Die eine besteht aus der neuen Kick 2.0, die andere aus der bisher üblichen Kick 1.3, die über das Menü, welches durch Drücken beider Maustasten während des Einschaltens erreicht werden kann. So weit, so gut. Daß trotzdem einige Programme streikten, liegt wohl an der unglaublichen Geschwindigkeitssteigerung.

Im Vergleich mit dem A2000/30 dürfte der A3000 wohl den besseren Schnitt machen.

Für welchen Rechner man sich letztendlich entscheidet, hängt von der eigenen Wunschvorstellung ab, bei der der Amiga 3000 wohl doch eher in Richtung »professionelle und kreative Arbeit mit Computern« geht.

Er wird seinen Platz finden, der “Traumcomputer“; wo und wie werden wir für Sie weiter beobachten.

(jb)

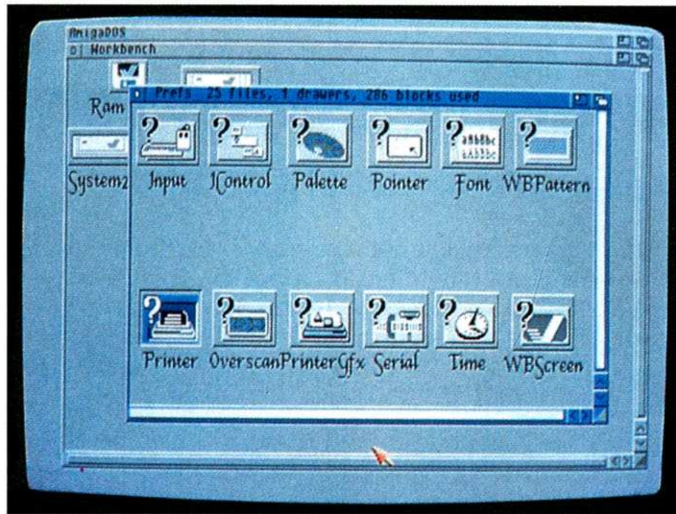


Bild 4. »Preferences« ist nur eines der Werkzeuge, die ihr Gesicht grundlegend gewandelt haben



Bild 5. Das im Tools-Verzeichnis befindliche »HDBackup« dient zur Sicherung von Festplattendaten auf Diskette oder Tape-Streamer

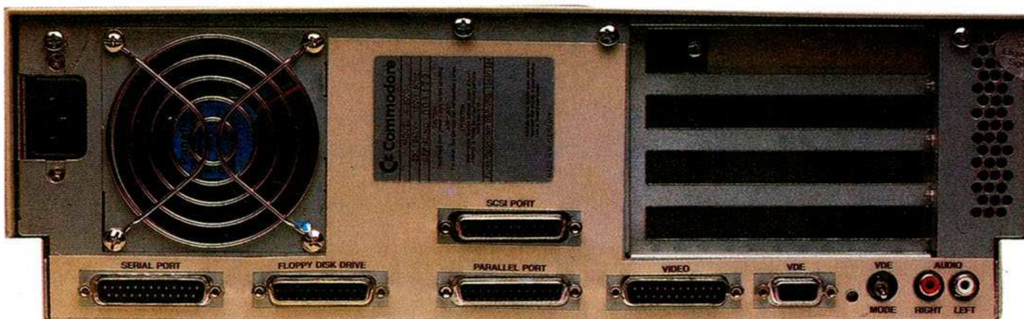


Bild 6. Die Rückseite des A3000 mit den Anschlußbuchsen, darunter VGA- und SCSI-Port

AMIGA DOS Info

Der Amiga 3000 ist das neue Flaggschiff der Amiga-Reihe. Ausgerüstet mit einem 68030-Prozessor und einem numerischen Co-Prozessor 68882 (68881 in der 16 MHz-Version) bürgt er für schnelle Befehlsabarbeitung, da er mit 25 MHz (16 MHz in Version 2) getaktet wird.

Das Chip-RAM besteht in der Grundausstattung aus 1 MByte, voll ausgebaut sind 2 MByte Chip-RAM adressierbar.

2 MByte Fast-RAM in der Grundausstattung können auf 18 MByte intern ausgebaut werden. Außer den bekannten Monitoren können auch Multiscan-Monitore über einen VGA-kompatiblen Anschluß angesteuert werden. Neben den normalen Port-Anschlüssen verfügt er noch über einen SCSI-Anschluß (Small Computer Standard Interface) für externe Festplatten oder Streamer. Die standardmäßige Festplatte hat eine Kapazität von 30 MByte; größere Kapazitäten sind optional. Der A3000 benötigt zum Start wieder eine Kickstart-Diskette, die jedoch auch ein Starten von Festplatte ermöglicht. Workbench-Screens und Windows wurden völlig überarbeitet und bringen ein ganz neues Erscheinungsbild zutage.

Der Preis des Computers wird zwischen 6000 und 7000 DM liegen.

Amiga 500 im neuen Gewand

Der Umbausatz MW-500

So gut und preiswert der Amiga 500 auch ist, so unpraktisch gestaltet sich manchmal die Arbeit mit dem 500er. Ursachen sind die nicht abgesetzte Tastatur und die seitlich am Gerät angebrachte Floppy. Abhilfe könnte hier das MW-500 der Firma Wenngatz schaffen.

Werfen wir zunächst einmal einen Blick auf den Lieferumfang des Umbausatzes. Neben einem flachen, PC-ähnlichen Gehäuse finden wir einen Bausatz für ein externes Tastaturgehäuse, einen Einbausatz für eine interne Zweitfloppy, eine Tastaturkabelverlängerung, diverses Anschlußmaterial für die Laufwerke sowie einen Schraubensatz, mit dem das Motherboard und die Floppies im Gehäuse fixiert werden. Ergänzt wird die Hardware durch eine detaillierte Umbauanleitung, mit deren Hilfe auch der Ungeübte den Umbau schnell und sicher bewältigen kann.

Da nicht gelötet werden muß, ist kein Spezialwerkzeug erforderlich, so daß einige Schraubendreher, eine Spitzzange und ein Teppichmesser für den Umbau ausreichen.

Allerdings existieren zwei Versionen des Umbausatzes. Wir haben für Sie die Version getestet, der ein zweites Laufwerk sowie die benötigte Zusatzplatine beiliegt.

Das Geheimnis der vierten Schraube...

Durch diverse vorangegangene Selbst- und Umbauten verschiedener Geräte gewitzt, haben wir uns zunächst einmal mit der aus einigen Din-A4-Blättern bestehenden Umbauanleitung auseinandergesetzt. Komplett in Deutsch gehalten, verzichtet die Anleitung auf Sätze wie "...anschließend legen Sie Formteil 23c/3b in die dafür vorgesehene Nut des Gehäuseunterteils 28d-4 um..." und hebt sich durch eine klare Step-by-step-Gliederung hervor. Zweckmäßig beginnt der Umbau mit dem Zerlegen des A 500. Nach dem Entfernen des Gehäuses wird zunächst die Tastatur abgenommen und mit dem ehemaligen »Df0:« beiseite gelegt. Anschließend werden die Abschirmung entfernt und die Schrauben gelöst, die das Motherboard im Gehäuseunterteil fixieren.



Bild 1. Der Lieferumfang des MW-500 Systems

Bitte unbedingt an dieser Stelle beachten: Berühren Sie die Lötseite des Motherboards möglichst nicht mit den Fingern, wenn Sie nicht vorher für entsprechende Erdung gesorgt haben!

Nach dem Entfernen des Motherboards hat das alte Gehäuse nun ausgedient und kann zum Beispiel als Terrarium neuen Verwendungszwecken zugeführt werden...

Der Einbau in das MW-500-Gehäuse gestaltet sich recht einfach. Zunächst sollte das Gehäuseunterteil für den Einbau der Floppies und des Motherboards vorbereitet werden. Zu diesem Zweck sind einige Push-Out-Blenden, je nach Bedarf, herauszudrücken. Die entstehenden Kanten können dann mit dem Teppichmesser geglättet werden. Falls vorhanden, wird so Zugang zum Zweitlaufwerk geschaffen. Die zweite Blende öffnet die linke Seite des Gerätes für die Herausführung der beim Umbau notwendigen Portverlängerung, die dem von uns getesteten Umbausatz beilag. Als erstes sollten nun die beiden Floppies in die dafür vorgesehenen Aufnahmeschächte montiert werden. Sollten Sie über Floppies verfügen, deren Bus und Stromversorgung gesteckt ist, brauchen Sie diese anschließend nur mit dem beigefügten Kabel anzuklemmen. Zuvor jedoch gilt es, das Motherboard in die dafür vorgesehene Aufnahme zu plazieren und mit dem neuen Schraubensatz zu fixieren. Hier klärt sich auch das eingangs angesprochene Geheimnis der vierten Schraube. In dem uns vorliegenden Prototyp waren einige der zum Befestigen des Motherboards notwendigen Schraubenlöcher um ein paar Millimeter verschoben. Dieser Fehler ist jedoch rechtzeitig erkannt und behoben worden, das heißt, daß die Umbausätze, die in den Verkauf gelangen, hundertprozentig passen.

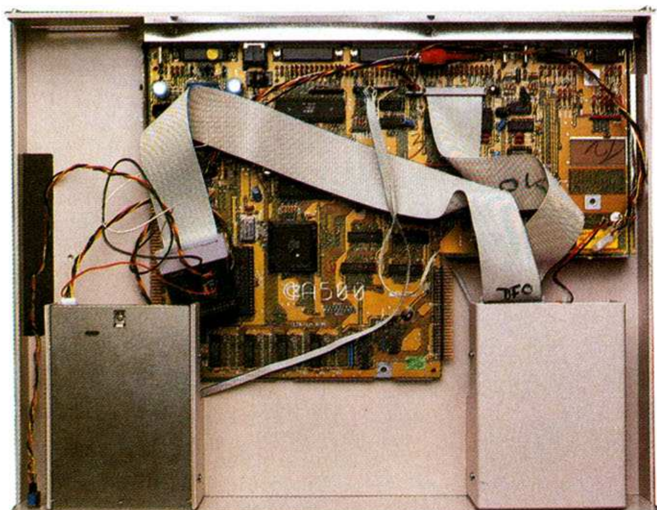


Bild 2. Das Motherboard mit Abschirmung und Floppies sind schnell umgebaut

Ist das Motherboard eingepaßt und mit den beiliegenden Schrauben fixiert, kann die Abschirmung wieder aufgesetzt werden. Eventuell vorhandene interne Speichererweiterungen können nun problemlos eingesetzt werden, Platz ist genug vorhanden.

Falls vorhanden, muß nun die zweite Floppy von innen am Floppy-Port angeschlossen werden. Die Verbindung wird dabei unter anderem über zwei Anschlußklemmen getätigt, die einer Prüfspitze ähneln. Nach dem Aufsetzen der Abschirmung kann nun die Hauptverkabelung beginnen. Mittels eines beiliegenden Buskabels werden die beiden Datenbusse der Floppies angesteckt. Die Spannungsversorgung übernimmt ein weiteres Steckkabel.

Nun braucht bloß noch das Tastaturverbindungskabel am Motherboard aufgesteckt zu werden, das die DIN-Buchse am Gehäusevorderteil mit der Hauptplatine verbindet. Um den internen Umbau zu vervollständigen, wird nun der Expansion-Port mittels einer Portverlängerung aus dem Gehäuse seitlich herausgeführt.

Ein kräftiger Druck schließt das Gehäuse, Teil 1 des Um-



Bild 3. Die Anschlüsse des A 500 liegen im MW-500 unverändert an gleicher Stelle

baus ist damit bewältigt. Bleibt noch die Tastatur, die in das neue Gehäuse eingepaßt werden muß. Hier ist auch der bisher einzige Kritikpunkt anzusetzen. Zunächst ist das Tastaturgehäuse zu öffnen und die Keyboard-Platine in die entsprechende Führungsnut einzulegen. Danach sind zwei keilförmige Abstandhalter einzusetzen, die die Tastatur im Endzustand in ihrer Position fixieren. Als nächstes ist die Verlängerung des Tastaturkabels anzuschließen, was mit Hilfe der bereits vorgefertigten Verlängerung schnell vonstatten geht.

Das anschließende Einschieben des Tastaturunterteils gestaltet sich dann etwas kniffelig, aber keine Sorge, der "Ruck", mit dem das Unterteil schließlich in seine vorgesehene Position rutscht, hat keine zerstörerischen Folgen und stellt sozusagen eine akustische Paßkontrolle dar.

Zukunftskompatibel

Leider bietet der Umbausatz keinen Platz für eine eventuell vorhandene Festplatte. Unter diesem Gesichtspunkt fallen zwei Aussparungen auf der Rückseite auf, die dazu dienen, die Erweiterungsbox des MW-500 anzuschließen. In dieser Erweiterungsbox werden laut Herstellerangaben dann zusätzliche RAM-Karten, Festplatten und ähnliches Platz finden. Alles in allem war der Umbau problemlos, die Anleitung ist detailliert und fängt denkbare Probleme gut ab. Auch optisch weiß das MW-500-System zu gefallen, und zumindest ein Teil des üblicherweise vorhandenen Kabelsalates läßt sich mit diesem System vermeiden. Nach unseren Erfahrungen benötigt der durchschnittliche Bastler zirka eine Stunde, bis der ehemalige A 500 im neuen Glanz erstrahlt. Zusammen mit der leider momentan noch nicht verfügbaren Erweiterungsbox haben wir es mit einem sinnvollen Umbausatz zu tun, der sich durch einfachen Aufbau und Robustheit sicher bewähren wird. Speziell Programmierer und Vielschreiber werden die abgesetzte Tastatur zu schätzen wissen.

(mm)

AMIGA DOS Blitzlicht

Name: MW-500

Hersteller: Computer & Zubehör, Miky Wenngatz

Vertrieb: Computer & Zubehör, Miky Wenngatz, 8031 Gilching, Jägerweg 31 Tel.: 08105/24540

Preis: 299,– DM + zweites Laufwerk

Besonderheiten: Erweiterungsbox optional

Positiv:
- abgesetzte Tastatur
- gelungenes Design
- hohe Funktionalität
- durchdachtes Konzept
- detaillierte Anleitung
- unproblematischer Umbau
- geringer Zeitaufwand
- hohe Paßgenauigkeit

Negativ:
- Netzteil nicht integriert

Bernd Rudolf

Vidi-Amiga

Digitalisieren in Echtzeit – Geschwindigkeit ist keine Hexerei

Mit Vidi-Amiga ist das Angebot an Digitizern für den Amiga wieder um ein Exemplar reicher geworden. Daß es sich hier um den Digitizer für den Amiga schlechthin handelt, merkt man jedoch erst auf den zweiten Blick.

Wenn man erwartungsvoll den Karton öffnet, der die Wunder-Hardware enthält, und ans Installieren geht, wird man herb enttäuscht, falls man nicht Besitzer eines Amiga 500 ist. Das schlichte Gehäuse des Vidi-Amiga ist so ungünstig geformt, daß es beim Anschluß an den Parallel-Port des Amiga 2000 den RGB-Ausgang blockiert. Entweder muß man einen monochromen Monitor verwenden, oder ein Verlängerungskabel muß her. Notfalls tut es auch ein T-Switch. Nach einer kleinen Lötaktion kommt das zweite Problem: Der Stecker für die zusätzliche Spannungsversorgung über den zweiten Joystick-Port paßt aufgrund seiner Bauweise nicht auf den selbigen. Das zweite Verlängerungskabel ist fällig! Wenn man jedoch diese Hürden genommen hat, wird man von einem Digitizer der Spitzenklasse überzeugt.

Venit, vidit, vicit - Er kam, sah und siegte!

In der Grundversion wird der Vidi-Amiga als Schwarzweiß-Echtzeit-Digitizer angeboten, der eine Auflösung von 320x256 Punkten in 16 Graustufen unterstützt. Das ist nichts Besonderes, wird sich der eine oder andere sagen, doch die Software ist vom feinsten: Abgesehen von den üblichen Funktionen wie Laden und Speichern von Bildern werden hier Effekte ermöglicht, die dem Anwender fast keine Grenzen setzen. Die wichtigste Eigenschaft des Vidi-Amiga ist die Unterstützung des gesamten Speichers. Auf diese Weise ist es möglich, eine ganze Szene aus einem Film zu digitalisieren. Die Geschwindigkeit ist bis zu einer oberen Grenze frei wählbar. Für Individualisten ist es sogar möglich, im Einzelbildmodus ein Bild nach dem anderen zu digitalisieren, wenn die Höchstgeschwindigkeit immer noch nicht ausreicht. Hat man nun eine komplette Szene eingelesen, kann man sich problemlos ein einzelnes Bild herausfischen und manipulieren.
Eine sehr nützliche Hilfsfunktion stellt dazu immer jeweils 16 Bilder der Serie auf dem Bildschirm dar, aus denen man sich das gewünschte Bild durch Hin- und Herblättern der Seiten und Anklicken mit der Maus auswählen kann. Es ist dabei völlig gleichgültig, ob man sich nur für das eine Bild oder für die gesamte Szene interessiert. Nun lassen sich beliebig große Ausschnitte definieren, die man frei in irgendwelchen Bildern umherkopieren kann. Selbstverständlich läßt sich die eingelesene Szene jederzeit wieder animieren, schneller oder langsamer, je nach Wunsch. Die fantastischste aller Funktionen ist jedoch die Definition eines Fensters beliebiger Größe, die die Einblendung eines Bildausschnitts aus einem zweiten Digitalisiervorgang in die bereits bestehende Szene ermöglicht. Natürlich funktioniert das auch umgekehrt. "Umgekehrt" bedeutet in diesem Fall, daß der Bildausschnitt der bestehenden Szene erhalten bleibt und der Rest "drumherumdigitalisiert" wird. Kreativität ist gefragt.



Bild 1. Das Vidi-Amiga System mit Vidi-Chrome. In dieser Kombination erzielt man beste Ergebnisse

Welche Farbe hätten's denn gern?

Als Erweiterung für Vidi-Amiga ist ein Farbfilter inklusive Software erhältlich, mit der man hervorragende Ergebnisse erzielen kann (siehe Bild 1). Die Methode, mit Farbfiltern zu digitalisieren, ist bereits von anderen Digitizern her bekannt, sie übertrifft jedoch in diesem Fall jegliche Vorstellungen. Vidi-Amiga digitalisiert nacheinander erst den Rot-, dann den Grün- und Blauanteil des Bildes. Alles in Echtzeit, versteht sich. Jede Veränderung des Lichts wird sofort auf dem Bildschirm sichtbar und kann vor der Bestätigung des Bilds durch einen Tastendruck noch korrigiert werden. Ist das gewünschte Bild eingelesen, so wird es aus den drei Bildern (rot, grün und blau) schnell zu einem HAM-Bild zusammengesetzt. Es besteht sowohl die Möglichkeit, die drei Farbanteile einzeln zu speichern, als auch ein komplettes Bild zu speichern, das man mit einem "Malprogramm" weiter bearbeiten kann.
Vidi-Amiga verarbeitet jedes Video-Signal, das Sie ihm anbieten, sei es PAL oder NTSC etc. Helligkeit und Kontrast sind sowohl hardwaremäßig als auch softwaremäßig einstellbar, und alle Funktionen sind kurz und präzise in einer deutschen Anleitung erklärt.

(mm)



Bild 2. Farbdigitalisierung mit Vidi-Amiga und Vidi-Chrome. Das Ergebnis spricht für sich

AMIGA DOS Blitzlicht

Name: Vidi-Amiga
Hersteller: Rombo Limited
Vertrieb: PR8-Soft, Otto-Hahn-Straße 10, 8702 Estenfeld
Preis: 398,– DM/496,– DM
Besonderheiten: mit Programm Vidi-Chrome in Farbe

Positiv:
- übersichtlich
- vielseitig
- durchdachte Konzeption
- Festplatte wird unterstützt
- mit Vidi-Chrome auch Farbe
- brauchbare Effekte

Negativ:
- Anschluß an A2000 nur mit Verlängerungskabeln

! ACHTUNG AUFNAHME !



Der Sampler »Perfect Sound« im Test

Was ist klein und handlich, und wenn man das Programm lädt, wird geschossen? Für alle, die keine Antwort wissen, hier ist sie: »Perfect Sound«, der Sampler und seine Software.

Der Soundchip »PAULA«, seines Zeichens Disk-Jockey des Amiga, kann neben der "normalen" Tonerzeugung noch eines ganz gut: digitalisierte Sounds wiedergeben. Und so tummeln sich auf dem Markt eine ganze Menge Sampler, Sound-Editoren und ähnliches für den Hitschreiber in spe. Zur Gruppe der Sampler gehört auch »Perfect Sound«, der sich allerdings einen Spitzenplatz ergattern will. Wie und ob – das haben wir für Sie in Erfahrung gebracht.

Klein in den Ausmaßen – groß in der Leistung

Im Paket des Perfect-Sound-Systems findet man neben der Software-Diskette und dem Handbuch den eigentlichen Sampler. Die Ausmaße des Gerätes werden vor allem die in Erstaunen versetzen, die mit Elektronik an sich noch nicht viel zu tun hatten; ein kleines graues Metallkästchen enthält einen kompletten Stereo-Sampler. Für die Audioeingänge sind zwei Cinch-Buchsen für den rechten und den linken Kanal und eine 3,5-mm-Klinkenbuchse für Mikrofon vorgesehen. An der Amiga-Seite befindet sich ein 25poliger Sub-D-Stecker für den Parallelport, der serielle, an dem die Audiodaten übertragen werden, wird von Perfect Sound nicht benötigt. Das Kästchen mit der Elektronik ist recht stabil und liegt gut am Stecker an, ohne eine fehlerhafte Kontaktverbindung zu erzeugen.
Als Zuleitung zum Sampler empfehlen sich abgeschirmte einpolige Audiokabel, die von der jeweiligen Signalquelle direkt zum Gerät geführt werden. Beim Test mit einem CD-Player, einem Keyboard und einem Mikrofon zeigte sich, daß die Signale teilweise zu sehr übersteuert an den Eingang des Samplers gelangten und die Wiedergabe daher verzerrt wurde. Erst der Einbau eines Signalabschwächers führte zu besseren Ergebnissen. Das ist kein Problem des Samplers, sondern der Signalquellen. Dieses Problem taucht nicht zum ersten Mal beim Einsatz eines Samplers auf. Wer in der Audioelektronik nicht so versiert ist, wird sich über einen "schlechten" Sampler ärgern, ohne zu wissen, ob seine Audiogeräte nicht doch die Urheber der Störungen sind. Vielleicht wäre der Einsatz von Lautstärkereglern an den Samplern doch überlegenswert.
Der beste Sound-Digitalisierer nützt überhaupt nichts, wenn nicht ein Programm, also die entsprechende Software, den Datenfluß zwischen Digitalisierer und Prozessor (sowie Speicher und Peripherie-Bausteinen) steuert. Die ankommenden Signale werden ja in parallel liegende Bit-Informationen umgewandelt, beim Parallelport also in acht Bit, befinden sich aber lange noch nicht dort, wo sie genutzt werden können, also im Soundchip. Somit kommt jetzt das Programm Perfect Sound zum Zuge.

Digitizer + Software = Sampler

Es meldet sich auch gleich mit ein paar kernigen Revolverschüssen, einer Explosion und dem Urschrei eines unfreundlichen Herren. Wer das Programm mit voll aufgedrehten Volumenreglern, sei es an der Stereoanlage, sei es am Monitor, startet, ist selbst schuld, wenn ihm gleich zu Anfang das Trommelfell platzt. Aber immerhin zeigt der Sampler gleich, was er kann.
Das Programm des Perfect-Sound-Systems setzt voll auf Grafik; viele Funktionen des Programms sind über »Buttons«, also Schalter, die per Maus betätigt werden, zu erreichen. Der Rest, das sind auch nicht wenige Funktionen, wird wie gewohnt über eine Menüleiste erreicht.
Das obere Grafikfeld zeigt die Impulskurve des digitalisierten Sounds. Ein Textfeld links unten zeigt eine Liste mit den im Speicher befindlichen Samples an. Diese Samples können durch Anklicken mit der Maus gestartet und angehört werden. Auf der gegenüberliegenden Seite befindet sich ein Info-Fenster mit Informationen über das Sample sowie über den restlichen Speicher, der noch genutzt werden kann.
Die »Buttons« gliedern sich in folgende Funktionen auf: »Play Range« spielt einen mit der Maus ausgewählten Teil des gesamten Samples. »Loop Range« spielt diesen Teil in immer wiederkehrender Reihenfolge, also in einer Endlosschleife. »Stop« erklärt sich wohl von selbst. »Set Play Speed« erlaubt die Einstellung der Abspielgeschwindigkeit, »Find Loop Point« sucht automatisch einen Einstiegs- und Endpunkt für einen Instrumenten-Loop. Dabei wird das Sample auf optimale Länge eines Instruments untersucht und dieser eingestellt. Hilfreich ist das vor allem, wenn man bestimmte Instrumente isolieren will. »Delete Range« löscht einen eingestellten Bereich des Samples, »Paste«

speichert den eingestellten Bereich zwischen und erlaubt es, diesen zwischengespeicherten Bereich mit »Insert« an jeder beliebigen Stelle des Samples einzusetzen. »Copy range to new slot« dient zum Überspielen eines Samples in einen anderen Samplebereich, der im Programm als "Slot" gekennzeichnet wird. Diese Slots sind also nichts anderes als Samples, die parallel im Speicher stehen und nur zur Bearbeitung in den Speicher geholt werden. »Append slot to slot« verbindet zwei oder mehrere Samples (Slots) zu einem einzigen, jedoch mit der Einschränkung, die durch den verfügbaren Speicherplatz entsteht.

Diese Button-Funktionen sind als Einstellungen für die zu bearbeitenden oder aufzunehmenden Samples gedacht, während die Menüeinstellungen sich dem Sample direkt widmen. In diesem Menü sind auch die Funktionen zum Erstellen eines Samples, also zum Digitalisieren, zu finden. Das Sample kann vom linken, vom rechten oder von beiden Kanälen übernommen werden; wenn beide Kanäle genutzt werden, hat das Sample Stereoqualität.

Wenn man aus den digitalisierten Klängen Teile markiert (mit dem gesamten Sample muß ebenso verfahren werden), kann dieser Teil separat verändert werden. Neben Veränderungen, wie einer Spiegelung der Impulskurve (rückwärtslaufendes Band), kann hier auch die sogenannte Ramp-Funktion genutzt werden. Durch »Ramp Up« oder »Ramp down« kann ein An- oder Abschwelleffekt erreicht werden. Die digitalisierten Klänge können so langsam bis zur vollen Lautstärke hochgefahren werden, oder langsam ausklingen.

Sounds im CLI – Musik beim Formatieren

Funktionen, die das digitalisierte Klangwerk bearbeiten, sind mehr als genug vorhanden. Diese fertigen Samples können nun zwecks weiterer Verwendung abgespeichert werden, dabei unterscheidet Perfect Sound drei Abspeichervarianten:

1) Im normalen IFF-8SVX-Format wird der Sample von allen gängigen Sound-Editoren, die in der Lage sind, komplette Musikstücke aus digitalisierten Sounds zu erstellen, akzeptiert.

2) Das Dump-Format enthält nur die Rohdaten des Sounds.

3) Im Comp-Format wird eine komprimierte Version des Samples auf Diskette oder Festplatte abgespeichert. Diese komprimierten Samples belegen zwar nur die Hälfte an Speicherplatz, sind jedoch als "Soundspender" in Editoren nicht mehr zu benutzen, da der Klang sehr vermindert wird. Diese Art der Sample-Speicherung hat jedoch den Vorteil, nicht so speicherplatzfressend wie die IFF- oder Dump-Daten zu sein. Samples von zirka 3 Minuten Dauer entwickeln sehr schnell eine Länge von bis zu 1,7 MByte.

Zusätzlich zum Sampler-Programm befindet sich noch das Public-Domain-Programm »Sound« auf der Diskette. »Sound« ist ein Abspieler für Samples von der Shell oder dem CLI aus. Interessant ist hier vor allem die Möglichkeit, Samples als eigenen Task im Hintergrund laufen zu lassen.

Im Test wurde ein Musikstück mit »Perfect Sound« gesampelt und auf Harddisk (1,7 MByte) abgespeichert. Durch Einbinden der Zeile »RUN SOUND Musikname« wurde das Musikstück während der Shell im Hintergrund abgespielt. »Sound« läßt die Ausgabe auf beide Kanäle zu sowie als Parameter die Anzahl der Wiederholungen des Stückes (»Loop«) sowie die Sample Speed Rate, also die Geschwindigkeit. Die Zugabe dieses PD-Programms ist für alle die interessant, die sich selbst nicht in der Lage sehen, die Samples weiterzuverarbeiten, trotzdem aber nicht auf eine Nutzung des Samplers verzichten wollen.

Was ist dran am perfekten Sound?

Das Komplettpaket »Perfect Sound« bietet viele interessante Details zum Editieren und Digitalisieren von Klängen. Im Zusammenspiel mit CD-Player, Keyboard sowie einem Naturinstrument (Konzertgitarre mit eingebautem »PickUp«) erwies sich der Sampler als zuverlässiges Peripherie-Gerät. Die Software ist an einigen Stellen noch verbesserungswürdig; so blieben beim Aufbau von Requestern Textteile irgendwo in der Landschaft stehen, bei Umwandlungen von markierten Soundteilen hing sich "good old" Amiga irgendwo in den Tiefen der mathematischen Wunderbäume am nächsten Ast auf, so daß der berühmte Dreifingergriff einem Sample vorzeitig das Licht ausblies. Trotzdem ist Perfect Sound empfehlenswert: Weder Kabelsalat noch schlecht abgespeicherte Sounds waren zu bemerken.

Die Draufgabe des »Sound« ist ebenfalls zu begrüßen, vor allem weil die Anwendung dieses nützlichen Utilities nicht nur auf »Perfect Sound« beschränkt ist. Fazit also: Ein gut gelungenes Sound-Paket.

(jb)

AMIGA DOS Blitzlicht

Name: Perfect Sound Digital Sound Sampler

Hersteller: Sunrize Industries

Preis: stand bei Redaktionsschluß noch nicht fest

Anbieter: Fachhandel

Positiv:
- Übersichtliche Funktionen
- gute Qualität der digitalisierten Klänge
- kleine Abmessungen des Samplers
- Zoom-Funktion für Teilsamples

Negativ:
- Löschfunktion nur nach Markieren eines Ausschnitts
- Screen wird durch Text zerstört

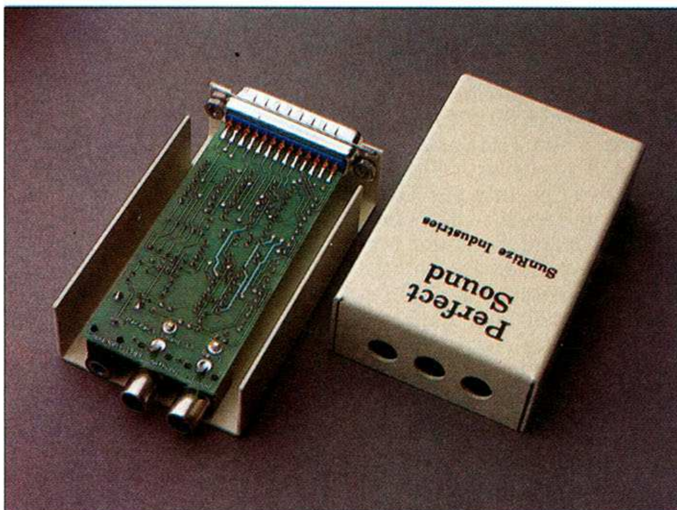


Bild 1. Erstaunlich, wie wenig Bauteile heute nötig sind, um Musik oder Sprache zu digitalisieren

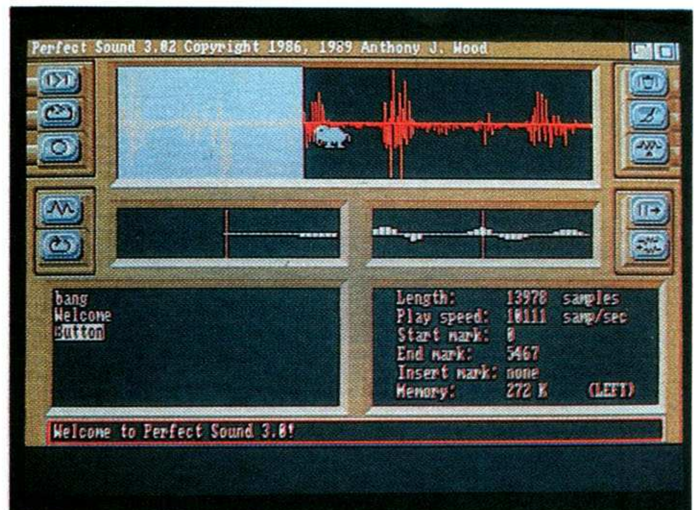
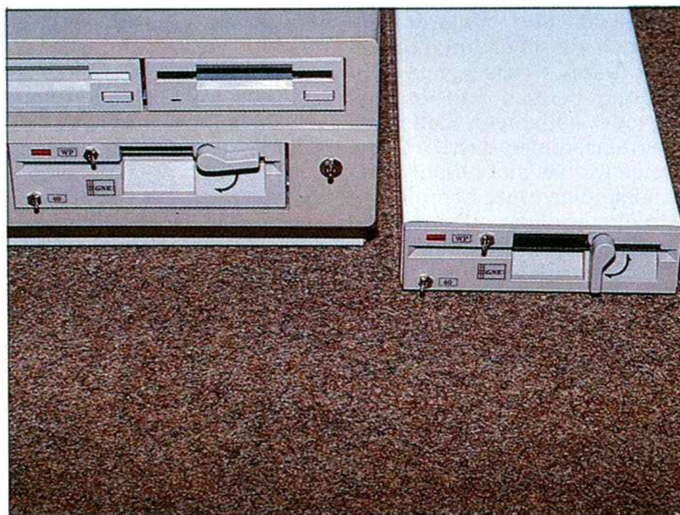


Bild 2. Im Programm können die digitalisierten Sounds auf vielfältige Weise manipuliert werden


Die 5,25-Zoll-Laufwerke der Firma GNE

Floppies für alle Fälle

Die GNE-Laufwerke mit Bootselektor

Mit der Zeit sammelt sich so einiges an Software bei einem Amiga-Besitzer an, die Kosten steigen für die 3,5-Zoll-Disketten. Bleibt also nur, auf die wesentlich günstigere 5,25-Zoll-Floppy umzusteigen. Und die sollte möglichst viel können.

Die beiden GNE-Laufwerke im 5,25-Zoll-Format unterscheiden sich von anderen ihrer Gruppe durch einen Bootselektor sowie durch mehrere eingebaute Funktionsschalter. Die Laufwerke besitzen einen Schalter mit der Aufschrift "Write Protect", der dafür sorgt, daß Disketten ohne Schutzaufkleber nach Betätigen des Schalters nicht mehr beschrieben werden können. Ein 40/80-Track-Umschalter erlaubt es, das Laufwerk auch für die PC-Karte nutzen zu können. Das 5,25-Zoll-Laufwerk wird als externes oder internes Gerät ausgeliefert, wobei das externe Gerät in einem stabilen Metallgehäuse untergebracht ist.
Die Stromversorgung wird bei beiden Geräten über den Floppy-Bus gewährleistet, der Bus ist bei beiden Laufwerken durchgeführt. Bei den Laufwerken handelt es sich durchweg um Teac FD55-Geräte, deren Ansteuer-Elektronik und Gehäuse von GNE in eigener Regie entwickelt wurden.
Der Einbau des internen Laufwerks gestaltet sich recht einfach, Montagematerial liegt dem Gerät bei.
Beide Laufwerke sind für Amiga-Besitzer, die Wert auf ein bißchen Komfort legen, durchaus zu empfehlen.

(jb)

AMIGA DOS Blitzlicht

Name: Profilaufwerk 5,25" intern (extern), wahlweise DF1: oder DF2:
Hersteller: Grebe Neumann Elektronik, Am Stein 10 (Hochstr.1), 5419 Raubach, Tel.:02684/5566 (5572)
Preis: intern DF1: 239,– DM, DF2: + Modifikationsplatine 259,– DM, extern 279,– DM, mit Bootselektor 289,– DM

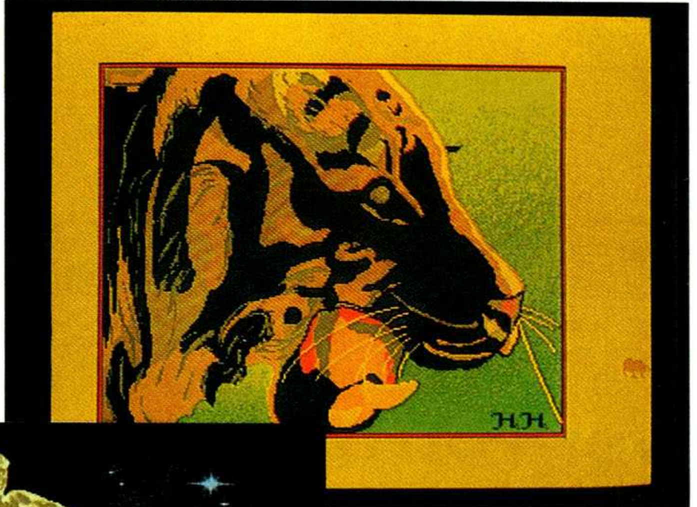
positiv:
- eingebauter Write-Protect-Schalter
- mit Bootselektor
- leichter Einbau

negativ:
- nicht negativ, aber überlegenswert: längeres Kabel für Bootselektor



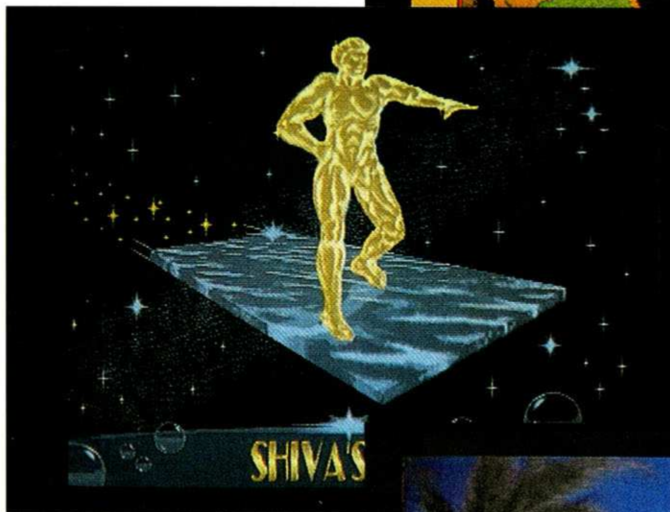
Pixel-Panorama

Die Seite für aktive Amiga-Grafikkünstler meldet sich wieder zur Stelle, und es gibt wieder Buntes und Interessantes zu begutachten. Aber sehen Sie selbst.



Tiger von Helmut Haslinger

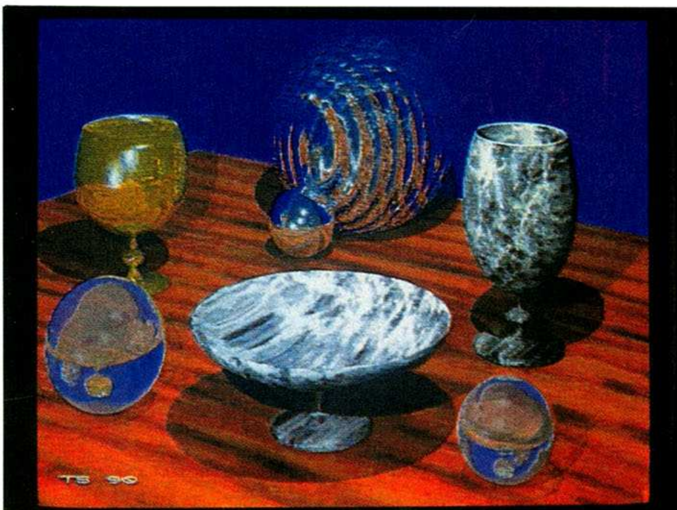
Das feine AMIGA-DOS-Logo, das diese Seite ziert, ist das Werk von Christian Mayer. Uns hat das Bild so gut gefallen, daß wir es Ihnen nicht vorenthalten wollten.
Tiger einmal anders. Helmut Haslinger hat sich dieses Tieres angenommen und ein expressionistisch coloriertes Portrait dieser Großkatze eingeschickt.
Die Vision einer goldenen Shiva auf einem galaktischen Surftrip hat sich Oliver Schafeld aus Reken ausgedacht.
Lukas Büter hat ein Bild seines Traumlandes kreiert. Ganz ehrlich, würden Sie nicht auch gerne einmal unter diesen Palmen liegen?
Den Abschluß bildet "Vessels", eine Raytracing-Vision von Thomas Schäfer, die für sich selbst spricht.



Oliver Schafelds Shiva's Surf



Lukas Büters Traumland



Vessels, ein Raytracing-Bild von Thomas Schäfer

Nach wie vor steht das Pixel-Panorama Ihren Einsendungen offen. Also, wenn Sie gute Amiga-Bilder zeichnen: Eine Diskette formatieren, Bilder draufkopieren, vermerken, welches Grafikformat und Programm verwendet wurde, und an folgende Adresse schicken:

**DMV - Verlag, Redaktion
AMIGA DOS
Kennwort Pixel
Postfach 250
3440 Eschwege**

Für Bilder, die wir im Rahmen des Pixel-Panoramas veröffentlichen, winkt ein Dankeschön in Form eines Software-Paketes im Wert von zirka 100 DM. (hs)

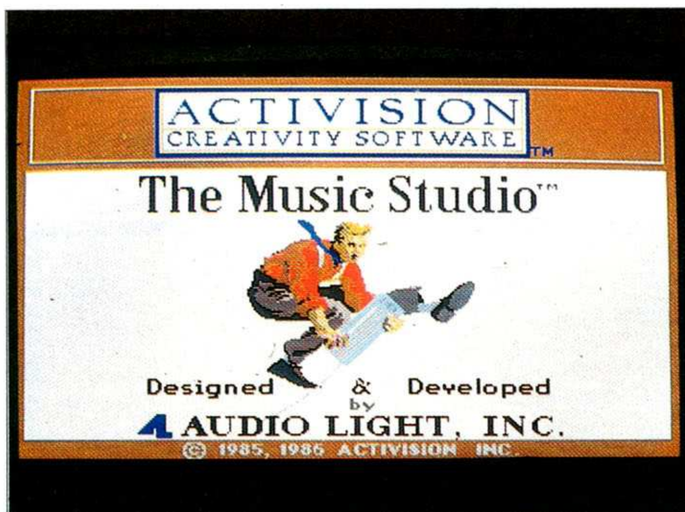


Bild 1. Das »Music-Studio« leitet ein ganzes MIDI-Orchester



Bild 2. Der »Sound-Tracker« ist sozusagen der Urvater der Sound-Editoren

Der Ton macht die Musik

Musikprogramme für jedermann

Mögen Sie auch Musik während der Arbeit oder dem Lernen? Laufende Radios bei den Schularbeiten oder im Büro haben schon viele Gelehrte auf den Plan gerufen, die dies für schädlich halten. Nun ja, die Eisenbahn mußte zu Anfang mit den gleichen Problemen kämpfen; Computer werden immer noch vielerorts verdammt. Wir Amiganer mögen es besonders gefährlich. Erstens arbeiten wir mit einem Computer, und zweitens können wir uns im Multitasking die schönsten Musikstücke anhören. Arme(r) Amiga-Besitzer(in).

Warum ist der Amiga ein so erfolgreicher Musiker unter allen Computern? Nun, zuerst sah es gar nicht so gut mit der Soundprogrammierung aus. Nur wenige Programme vermochten dem Soundchip einigermaßen vernünftige Klänge zu entlocken. Das Zauberwort hieß MIDI (Musical Instrument Digital Interface). MIDI stand und steht für Datenaustausch zwischen Musikinstrumenten und Computer – sofern beide über eine entsprechende MIDI-Schnittstelle verfügen. Hier ist auch der Hauptgrund zu finden, warum Computer in puncto Musik immer "die zweite Digitalgeige" spielen werden. Kein Programm kann den Klang eines Musikinstrumentes ersetzen. Allerdings können nicht alle Musikinstrumente mit einem MIDI-Interface ausgestattet werden (können Sie sich eine programmgesteuerte Maultrommel vorstellen?). Vor allem Tasteninstrumente, die über eine elektronische Tonerzeugung verfügen, werden sehr oft mit MIDI-Interfaces ausgestattet. Allerdings findet man auch schon Gitarren, Blasinstrumente (digitales Saxophon) oder Digi-Violinen, die sich hervorragend mit einem "Bitmonster" verstehen.

Amiga und Musik – Vorzüge und Nachteile

MIDI ist aus der modernen Musikwelt nicht mehr wegzudenken, für viele Profis gehört ein Computer mit MIDI-Interface heute schon zur Grundausstattung des Tonstudios – ein Muß, wenn man "den Hit" schreiben will.

So schön MIDI ist, beim Amiga gibt es da ein Handikap. Er verfügt nicht über ein eingebautes MIDI-Interface wie zum Beispiel sein Mitbewerber Atari ST. Dieses muß erst als Zubehör gekauft werden. Dann jedoch, mit der entsprechenden Software versteht sich, zeigt auch der Amiga, was er MIDI-mäßig draufhat.

Die entsprechende Software war allerdings am Anfang kaum zu bekommen, und wenn, dann mußten meistens Abstriche gemacht werden, die bei anderen Computern nicht notwendig waren. Musik, von Instrumenten gemacht und vom Computer gesteuert, das war ein Thema, was dem Amiga-Programmierer irgendwie nicht behagte – und das aus gutem Grund.

Der Amiga hat bei seiner Entstehung eine Gabe mitbekommen, die alle anderen Computer vor Neid erblassen läßt. Digitalisierte Tonfrequenzen werden in sehr guter Qualität an Stereo-Audioausgänge weitergegeben und können von dort direkt auf eine Stereoanlage gelangen. Wer gern mal die Multitasking-Fähigkeit des Amiga testen will, der soll sich eines der Abspielprogramme aus der Public Domain besorgen, seine Lieblings-CD digitalisieren und im Hintergrund abspielen lassen. Im Task 2 Rolling Stones, im Vordergrund eine Textverarbeitung, das ist doch schon was, oder? (Schöne Grüße an die hausinternen PC-Redaktionen – könnt Ihr das auch, Jungs?)

Der Soundchip des Amiga, PAULA genannt, kann zwar Wellenformen synthetisieren, viel besser allerdings liegen ihm (oder ihr?) digitalisierte Sounds. Was aber sind eigentlich digitalisierte Sounds?

Jeder von uns weiß, was ein Kassettenrecorder ist. Man schließt ihn an ein Radio oder ein Mikrofon an und nimmt im Nu alles auf, was einem gefällt, sei es die Hitparade oder ein Interview mit Prominenten. Diese Tonfolgen, also Gespräche, Geräusche, Musik, werden als Information auf einem Magnetband aufgezeichnet und können jederzeit wieder abgespielt werden. Im Grunde passiert nichts anderes als das eben Gesagte, wenn man vom Digitalisieren auf einem Computer spricht – mit einer großen Ausnahme.

PAULA, die Unermüdliche, wenn's um digitalisierte Sounds geht

Während die Aufnahme auf herkömmlichen Kassettenrecordern analog gespeichert wird, brauchen wir die Informationen für den Computer digital. Analog heißt, daß die Impulse, die den Sound bilden, mit allen Zwischenwerten vorhanden sind, während bei der Digitalisierung die Werte umgewandelt werden müssen. Bei der Digitalisierung wird jeder Impuls einer Zahl zugeordnet. Da beim Amiga nur 8 Bit (1 Byte) für die umgewandelten Impuls-

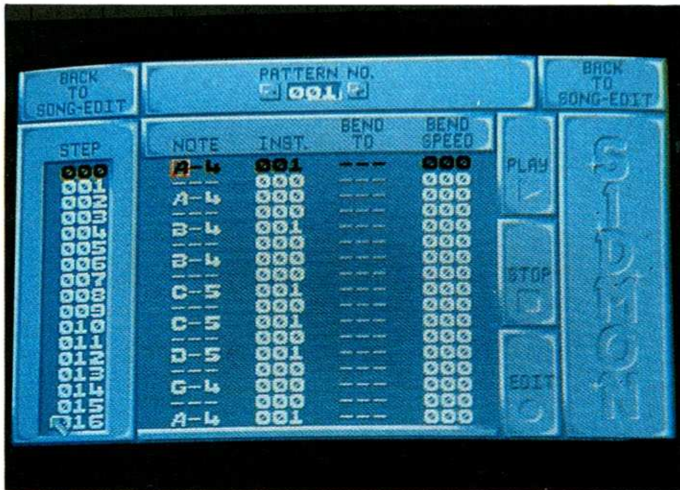


Bild 3. »SIDMON«, ebenfalls Sound-Editor, arbeitet in der neuen Version auch mit MIDI zusammen



Bild 4. Das »Mark II Sound System« enthält einen Scanner zur Fehlersuche

teile zur Verfügung stehen, reichen die Zuordnungszahlen nur von -128 bis 127. Dies sind genau 256 einzelne Zuordnungszahlen, in die der digitalisierte Impuls zerlegt werden kann. Da Tonschwingungen im positiven Bereich und im negativen Bereich stattfinden können, muß also hier eine Aufsplittung des Bytes in einen negativen und einen positiven Teil erfolgen.

Ob MIDI-Interface oder Sampler, Musik macht Spaß

Durch die Umwandlung der Tonfrequenzen in ganze Zahlen wird ein Teil der Töne quasi abgeschnitten, die Folge ist eine Verschlechterung der Tonqualität. Digitalisierte, vom Rechner wiedergegebene Tonfolgen sind daher kaum mit Tonbandabspielungen zu vergleichen. Hier kommt es auf die Qualität des Analog-/ Digitalwandlers an.

Diese A-/D-Wandler, die Tonfrequenzen in Digitalwerte umwandeln, nennt man im allgemeinen Sampler. Sampler bedienen sich meistens des parallelen Ports unseres Amiga, weil es hier gute Möglichkeiten zum Übertragen von Bytes zwischen Computer und Peripherie gibt – fragen Sie Ihren Drucker.

Die eingelesenen Digitaldaten müssen natürlich auch zu PAULA und von dort an die beiden Audioausgänge gelangen, sonst hört man nichts. Dies ist die Aufgabe der Software, die den Samplern beiliegt. Die Software sorgt für das Einlesen der Sounds vom Sampler, für das "Reinigen" derselben, indem Rausch-, Knack- oder andere Störimpulse angezeigt und entfernt werden können, und für Tricks, mit denen die Sounds verändert werden können (Modern Talking rückwärts – ein Genuß!).

Welche Möglichkeit für Sie in Betracht kommt, wenn Sie musikalisch sind (oder werden wollen), bleibt Ihnen überlassen. Wir stellen Ihnen nachfolgend noch ein paar Programme vor, die wir schon einmal getestet haben; nachfolgend finden Sie auch eine Marktübersicht über Musik-Soft- und Hardware. Und für die, die gerne selbst Musik programmieren wollen, haben wir anschließend noch eine Einführung in die Programmierung unter Assembler, beschrieben von zwei Fachleuten, nämlich den Programmierern von »Soundfactory«, einem der hier aufgeführten Soundprogramme.

Suchen Sie sich aus, was Ihnen liegt, der Amiga unterstützt Sie in jeder Hinsicht.

Inzwischen ist es gar nicht mehr so leicht, eine Trennung zwischen MIDI-, Sound-Editor- und Sampler-Programmen zu erkennen. Viele Sound-Editoren verfügen inzwischen auch über eine MIDI-Funktion.

Eine kleine Auswahl von Musikprogrammen

Um Ihnen einen kleinen Überblick zu geben, stellen wir Ihnen noch einmal bekannte Programme zum Thema Musik vor.

Für MIDI-Freunde fallen einem auf Anhieb zwei Programme ein; das eine eher für den Heimbereich, das andere für Profis und die, die es werden wollen. »The Music Studio« ist ein komplettes Kompositionsprogramm, welches einmal die Erzeugung von

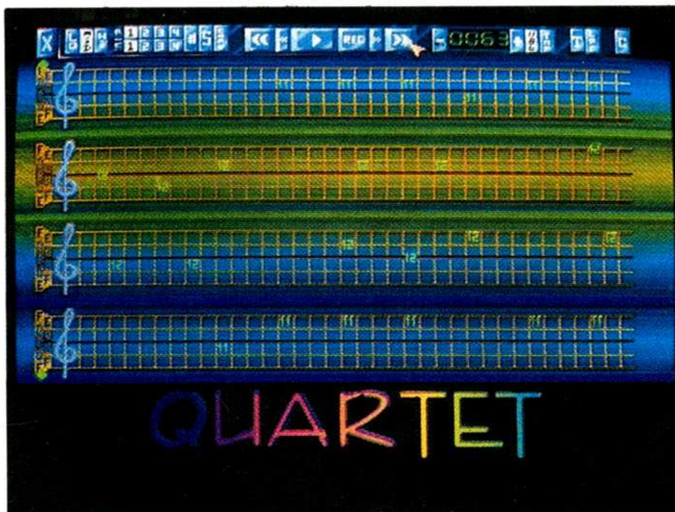


Bild 5. Ein etwas ungewohntes Bild bietet der Sound-Editor »Quartet«



Bild 6. »Quest I Texture« ist ein professioneller MIDI-Sequencer



Bild 7. »TFMX«, ein Profi-Sound-Editor, enthält eine komplette Makrosprache für Sounds

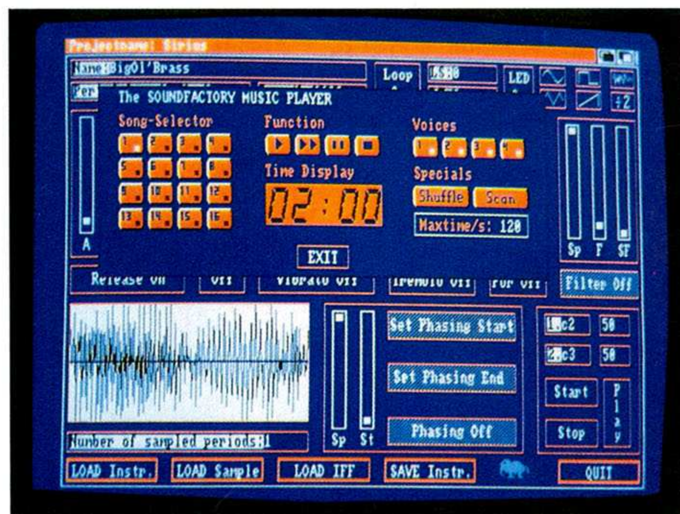


Bild 8. »Soundfactory« basiert auf einer eigenen Programmiersprache für Sounds

Wellenformen durch den Soundchip zuläßt, auf der anderen Seite die MIDI-Schnittstelle unterstützt. Hier liegt auch die Stärke des Programms. Während die Wellenformen des Soundchips etwas "geknödelt" wirken, funktioniert die Ausgabe auf die MIDI-Schnittstelle reibungslos. Welche Schnittstelle benutzt wird, ist gleich; »Music Studio« hat keine besonderen Ansprüche. Zu empfehlen ist das Programm jedem, der über ein MIDI-Instrument verfügt und komponieren will.

»Quest I Texture« ist ein professionelles MIDI-Programm, das vor allem die Musik-Profis ansprechen wird. Das Editieren der Musik geschieht über Tracks und Patterns, also Teilstücke einer Gesamtkomposition. Über die MIDI-Schnittstelle lassen sich mehrere Instrumente verwalten. Wer mit digitalisierten Sounds arbeiten möchte, ist mit Sound-Editoren am besten bedient. Der "Sound-Tracker", als "Urvater" der Sound-Editoren bekannt, brachte den Heimmusikern zum ersten Mal die Gelegenheit, Supersounds auf dem Amiga zu erzeugen und per Abspielroutine in eigene Programme einzubinden. "SIDMON" ist eine logische Weiterentwicklung dieser Programmart. Das Programm erlaubt das Editieren von digitalisiertem Sound in Patterns und Tracks, wobei jede Sample-Note, also der digitalisierte Ton, noch vielfältig beeinflußt werden kann. Außerdem sind Wellenformen nach eigenem Wunsch editierbar und in das Musikstück einbindbar. Die neueste Version von "SIDMON" kann darüber hinaus ebenfalls MIDI-Instrumente über ein entsprechendes Interface ansteuern.

In die gleiche Rubrik fallen auch die Programme »Mark II Sound System«, »TFMX« und »Soundfactory«. »TFMX« verfügt über eine implementierte Makrosprache, die es erlaubt, die eingelesenen Samples nach Belieben zu verändern. Das Programm ist ebenfalls MIDI-fähig. »Soundfactory« geht einen neuen Weg. Eigene Wellenformen oder fertige Samples werden in einer eigenen Programmiersprache editiert und danach wie bei einem Assembler zu einem ablauffähigen Programm gebunden. Diese Musikprogramme können ebenfalls in eigene Programme eingebunden werden. Wer nicht selbst musikalisch ist, muß dennoch nicht auf guten Sound verzichten. Sampler aller Art sind bei vielen Firmen schon recht preiswert zu bekommen. Hier sollte man darauf achten, ob man es mit einem Mono- oder einem Stereo-Sampler zu tun hat, zu sehen ist dies an der Anzahl der Audio-Eingänge. Stereo-Sampler "verbrauchen" mehr Speicher als Mono-Sampler, deshalb sollte man auch im Hinblick auf Speicherplatz nicht knausern. Bei Stereo-Samplern lohnt sich der Einsatz eines CD-Players, allerdings können 3-Minuten-Samples bis zu 2 MByte Speicher beanspruchen.

Das war unser kleiner Ausflug in die Welt der Musikprogramme; wer selbst programmieren möchte, sollte sich unseren Artikel "Ohne Bit kein Hit" durchlesen, der die Grundzüge der Soundprogrammierung näherbringt.

(jb)



Bild 9. Sound-Sampler wandeln Musik, Sprache oder Geräusche in Digitalinformationen um

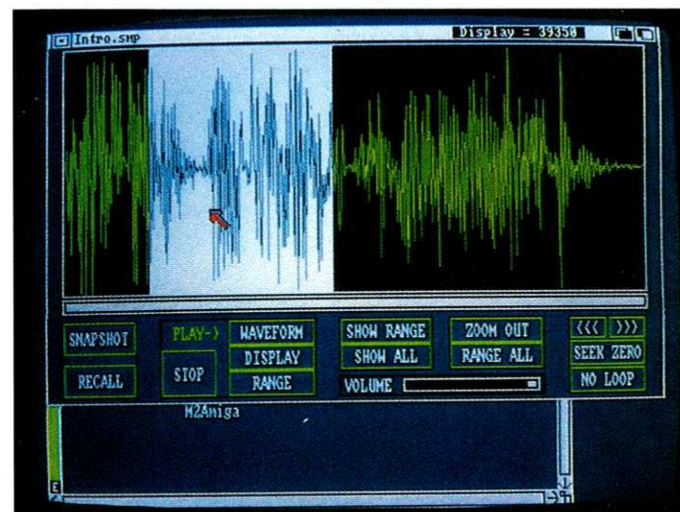


Bild 10. Die Software eines Sound-Samplers erlaubt vielfältige Änderungen

Marktübersicht Musik-Soft- und Hardware

Ein kleiner Überblick über Programme und Peripherie

Musik auf dem Amiga ist keine Schwierigkeit – wenn man weiß, welches Programm oder welche Hardware man nutzen will. Auf dem Markt für Computerzubehör findet sich allerhand, was zum Thema Musik gehört.
Wir haben an dieser Stelle verschiedene Programme und Geräte zusammengetragen, die Ihnen einen kleinen Überblick geben und Ihnen eventuell die Kaufentscheidung erleichtern können. Anschriften der entsprechenden Firmen finden Sie hier ebenfalls.

Um Ihnen die Übersicht zu erleichtern, haben wir Soft- und Hardware getrennt. Programme, die über eine Erweiterung verfügen, zum Beispiel Sound-Editoren mit MIDI-Teil, werden nur ihrer Hauptthematik zugeordnet.
Unsere Übersicht kann naturgemäß nicht alle Programme berücksichtigen, wir haben Ihnen daher nur die bekanntesten zusammengestellt.

Sound-Sampler und Digitizer

In die Kategorie der Sound-Sampler und Digitizer fallen alle Geräte, die Töne, Geräusche, Musik oder Sprache von Audio-Geräten (Tonband, CD-Player, Plattenspieler, Mikrofon) in digitale Impulse für den Amiga umwandeln. Hier gibt es zum Beispiel folgende Geräte:

- DeLuxe Sound V2.8, Hagenau Computer GmbH,
  DM 198,– (Amiga 1000)
  DM 228,– (Amiga 500/2000)
- Amiga Sound-Sampler, Data 2000,
  DM 79,– (Mono)
  DM 98,– (Stereo)
- Profi-Sampler, Omega Datentechnik,
  DM 139,– (Mono)
  DM 215,– (Stereo)
- Amiga Pro-Sampler-Studio + Datel Jammer, Eurosystems
  DM 169,–
- Sound-Sampler und Sampler-Studio, Alcomp GmbH,
  DM 79,– (Sound-Sampler, Hardware)
  DM 69,– (Sampler-Studio, Software)
  DM 129,– (Hard- und Software)
- Sound-Digitizer, Vesalia
  DM 89,–
- Sampler + Software, Computing (Österreich)
  öS 1390,– (zirka DM 198,–)
- Golem Sound II, Kupke Computertechnik,
  DM 189,– (Hardware)
  DM 149,– (Golem Sound-Mashine, Software)
  DM 299,– (Soft- und Hardware)
- Perfect Sound, (Hersteller Sunrize Industries)
  DM 150,– (zirka, Anbieter noch nicht bekannt)
- AMAS (Hersteller Microdeal)
  DM 200,– (zirka, Anbieter noch nicht bekannt)

Sound-Edit-Programme

Die Sound-Editoren verbinden gesampelte Sounds zu komplexen Musikstücken. Hierbei spielt die Noteneingabe eine untergeordnete Rolle, die Eingabe geschieht durch Numerierung einzelner Teilstücke oder Samples. Andere Editoren wiederum verwenden eine komplette eigene Musiksprache.

- The Ultimate Soundtracker, EAS,
  DM 129,– (zwei Disketten, Programm und Samples)
- Sidmon, Turtle Byte Software
  DM 89,– (zwei Disketten, Programm und Samples)
- Mark II Soundsystem, Cachet
  DM 49,– (zwei Disketten, Programm und Samples)
- TFMX, Demonware,
  DM 130,– (drei Disketten, Programm und zwei Sample-Disks)
- Sound-Factory, Profiteam-Software
  DM 129,– (zwei Disketten, Programm und Samples)

MIDI-Programme und Interfaces

MIDI-Interfaces bilden die Brücke vom Amiga zu den professionellen Musikinstrumenten und Soundgeräten. Über eine MIDI-Schnittstelle können mehrere Instrumente gesteuert werden, dabei ist ausschlaggebend, über wie viele Anschlüsse das MIDI-Interface verfügt. Bei den Anschlüssen unterscheidet man zwischen IN (empfangene Daten), OUT (gesendete Daten) und THRU (Daten, die beim Empfänger nur "durchgeschleust" werden, um sie an weitere Endgeräte zu senden). MIDI-Software übernimmt die Steuerung der Interfaces und ist die Schnittstelle zum Benutzer.

- DeLuxe MIDI, Hagenau Computer GmbH
  DM 98,– (Interface, 1 * IN, 2 * OUT, 1 * THRU)
  DM 128,– (Profi-Midi)
- MIDI-Interface, Data 2000,
  DM 89,– (Interface, 1 * IN, 4 * OUT, 1 * THRU)
- MIDI Master, Eurosystems
  DM 99,– (Interface, 1 * IN, 3 * OUT, 1 * THRU)
  DM 49,– (MIDI Music-Manager, Software)
  DM 120,– (Hard- und Software)
- MIDI Interface, Alcomp GmbH
  DM 89,– (Interface)
- Winner MIDI, Vesalia
  DM 89,– (Interface)
- MIDI, HK-Computer
  DM 89,– (Interface, 1 * IN, 2 * OUT, 1 * THRU)
- MIDI Interface, Computing (Österreich)
  öS 799,– (zirka DM 115,–, Interface)
  öS 999,– (zirka DM 150,–, Software)

MIDI-Software

- Quest I Texture, Sound Quest, zirka DM 250,–
- Quest II Texture, Sound Quest, zirka DM 500,–
- Music Studio, Activison, zirka DM 130,–
- MIDI Music-Manager, Eurosystems, DM 49,–
- Music X, Micro Illusions, zirka DM 500,–

Anschriften der Anbieter

- Alcomp GmbH, Glescher Weg 22, 5012 Bedburg, Tel.: 02272 / 2093
- Cachet, Ostendstr. 32, 7524 Oestringen, Tel.: 0725 / 322411 (Mark II Soundsystem)
- Computing, Schulgasse 63, A-1180 Wien, Tel.: 0222 / 4085256
- Data 2000, Stresemannstr. 11-16, 5800 Hagen 1, Tel.: 02331 / 23290 (31272)
- Demonware, Strahlenberger Str. 125a, 6050 Offenbach (TFMX)
- EAS Software GbR, Ferdinandstr. 16, 4630 Bochum 1, Tel.: 0234/34307
- Eurosystems, Hühnerstr. 11, 4240 Emmerich, Tel.: 02822 / 45589
- Hagenau Computer GmbH, Alter Uentroper Weg 181, 4700 Hamm, Tel.: 02381/880077
- HK-Computer, Bonner Str. 37, 5000 Köln 1, Tel.: 0221 / 311606
- Kupke Computertechnik, Burgweg 52a, 4600 Dortmund 1, Tel.: 0231 / 818325-27
- Musik- und Grafiksoftwareshop, Wasserburger Landstraße 244, 8000 München 82, Tel.: 089 / 4306207 (Quest I + II)
- Omega Datentechnik, Junker Str. 2, 2900 Oldenburg, Tel.: 0441/71109
- Profiteam Software, Wildermannstr. 62, 4350 Recklinghausen, Tel.: 02361 / 652229
- Turtle Byte Software, Postfach 830110, 5000 Köln 80, Tel.: 0221/602216
- Vesalia, Industriestr. 25, 4236 Hamminkeln, Tel.: 02852 / 1068



Thomas Kolbe / Zuheir Urwani

Ohne Bit kein Hit

Wissenswertes zur Programmierung von Musik

Der Amiga ist ein Rechner mit phantastischen Grafik- und Soundfähigkeiten. Doch was nutzen die tollsten Fähigkeiten, wenn man nicht weiß, wie man sie programmieren kann. In diesem Artikel sollen Ihnen die Grundlagen für die Programmierung von Soundeffekten und einfachen Melodien an mehreren Programmbeispielen gezeigt werden, so daß Sie anschließend in der Lage sein sollten, Sound und Musik in Ihre eigenen Programme zu bringen.

Um die Erzeugung von Klängen verstehen zu können, muß als erstes die Frage "Woraus besteht eigentlich ein Klang?" gestellt werden. Ein Klang ist im Grunde nichts anderes als Schwingungen, die von einem Instrument oder einer anderen Geräuschquelle erzeugt und von der Luft übertragen werden. Solche Schwingungen werden in der elektronischen Musik mit Hilfe von Schaltkreisen erzeugt und über einen Lautsprecher ausgegeben. Jede Schwingung oder jeder Ton wird durch einige wenige Parameter charakterisiert. Diese Parameter werden repräsentiert von der Frequenz, der Lautstärke und der Schwingungsform.



Drei Parameter erzeugen eine Schwingung

Die Frequenz bestimmt, wie hoch der Ton klingt. Sie wird in Hertz (Hz) angegeben, was die Anzahl der Schwingungen pro Sekunde bezeichnet. Je mehr Schwingungen pro Sekunde ein Ton hat, desto höher klingt er. Der Hörbereich umfaßt aber nur ein begrenztes Frequenzspektrum, welches beim Menschen bis etwa 18 kHz reicht.



Die Lautstärke gibt die Amplitude der Schwingungen, also den Schalldruck des Tones an. Der Verlauf der Lautstärke vom Einsetzen des Tones bis zum völligen Ausklingen wird auch als Hüllkurve bezeichnet. Bei einem Klavier setzt der Ton sofort mit voller Lautstärke ein und klingt dann langsam aus, während bei einer Flöte die Lautstärke relativ langsam anwächst und fast abrupt absinkt. Die Schwingungsform bestimmt die Klangfarbe eines Tons; sie kann eine einfache periodische Sinus-, Rechteck- oder Dreieckschwingung oder aber auch eine komplizierte, nicht genau definierbare Schwingung sein; zum Beispiel ein Rauschen oder ein Klirren.

Im Amiga werden die Schwingungsformen digital in Form von 8-Bit-Zahlen gespeichert. Der Vorteil ist dabei, daß jede erdenkliche Schwingung als eine Folge von Zahlen im Speicher gehalten und abgespielt werden kann. Die Schwingung wird dazu in gleichmäßigen Abständen abgetastet und der jeweils aktuelle Wert (Sample) als digitaler Wert abgelegt (siehe auch Abb. 1). Bei der Ausgabe werden diese so erzeugten Zahlenwerte wieder in die den Werten entsprechende Spannungen umgewandelt. Dabei kann aber nicht mehr die Qualität der Originalschwingung erreicht werden, da durch die vorherige Analog-/Digitalwandlung (A/D) "Treppen" entstanden sind. Die Klangqualität eines Tones hängt somit von der Auflösung und der Frequenz (Sampling-Rate) der Digitalisierung ab. Hier gilt: Durch eine größere Auflösung erhält man feinere Stufenunterschiede, und durch eine größere Sample-Rate werden mehr Zwischenwerte aufgenommen. Dies wiederum bedeutet, daß eine viel bessere Annäherung an den Originalton stattfindet und dieser somit auch besser wiedergegeben werden kann. Beim Amiga ist die Auflösung auf 8 Bit festgelegt, was bedeutet, daß die Sample-Daten Werte von -128 bis 127 annehmen können. Die Sampling-Rate ist nach oben auf 28867 Hz beschränkt.

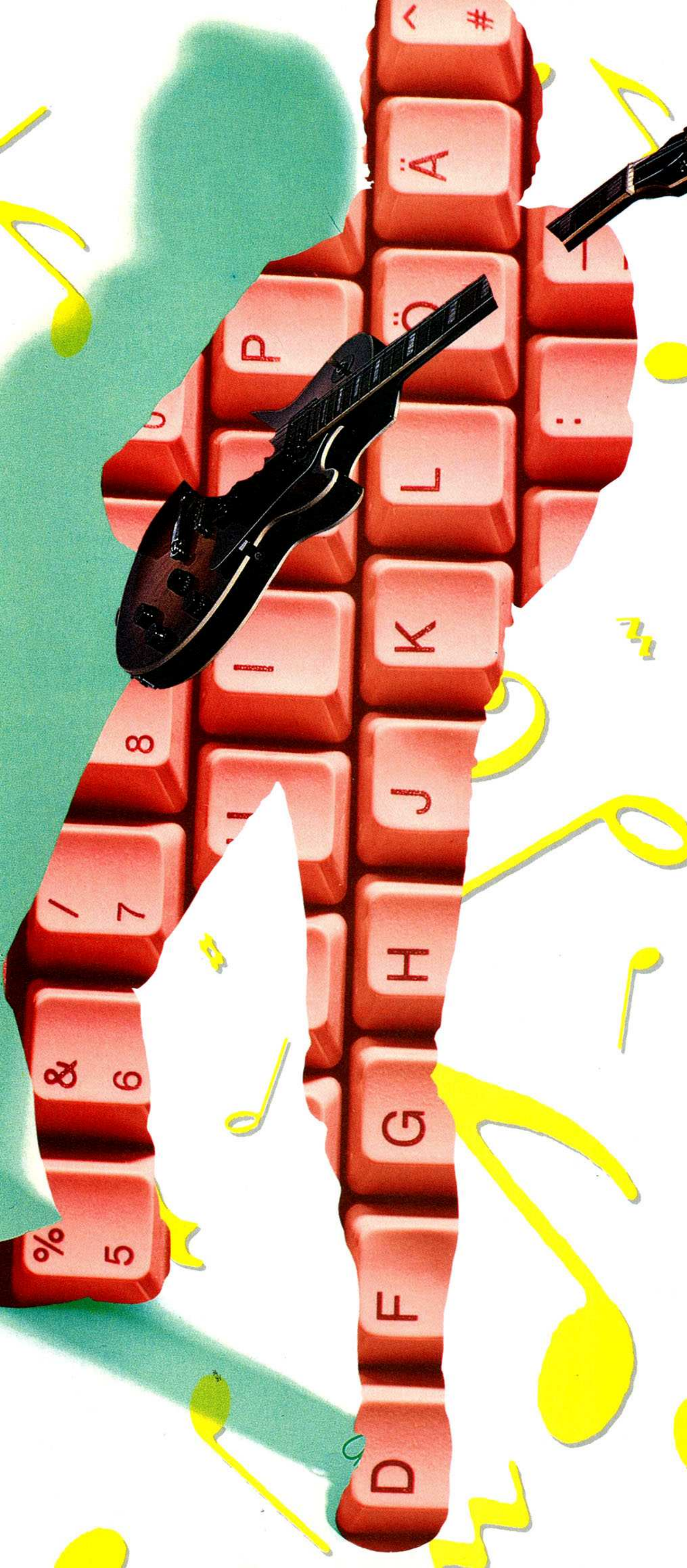
Musikalisches Programm oder programmierte Musik – was ist Ihnen lieber?

Doch kommen wir nun zu der konkreten Programmierung der Amiga-Hardware. Die Steuerung der Musikausgabe geschieht über Register, die an normalen Adressen im Speicher des Amiga eingeblendet sind. Tabelle 1 zeigt diese Register mit deren Bezeichnungen und Adressen.

Wie man unschwer erkennen kann, sind die Register in vier gleiche Gruppen unterteilt, die sich nur durch die Indizes 0-3 in den Bezeichnungen unterscheiden. Jede dieser Gruppen ist für einen Soundkanal zuständig, denn der Amiga ist in der Lage, vier Sounds gleichzeitig und völlig unabhängig voneinander auszugeben. Dabei gilt wegen der Stereofähigkeit des Amiga: die Kanäle 0 und 3 werden über den linken, Kanal 1 und 2 über den rechten angeschlossenen Lautsprecher ausgegeben.

Um nun einen Ton über einen Kanal auszugeben, muß man zuerst wissen, wo die Sample-Daten des Tons im Speicher liegen. Der Amiga verfügt nicht über vordefinierte Klänge, man muß daher auch einfachste Schwingungsformen (zum Beispiel eine Rechteckschwingung) selbst definieren und im Speicher ablegen. An dieser Stelle muß erwähnt werden, daß diese Sample-Daten unbedingt im Chip-RAM des Amiga stehen müssen, denn wie bei der Grafik auch, können die im Amiga für den Sound und die Grafik zuständigen Custom-Chips nur auf die 512 kBytes Chip-RAM, welche im Adreßbereich von $000000 bis $07FFFF liegen, zugreifen. Neuere Amiga-Modelle sind bereits mit 1 MByte Chip-RAM ausgestattet, hier steht dann natürlich mehr Speicherplatz zur Verfügung, um Sound- und Grafikdaten unterzubringen.

Nehmen wir an, daß wir eine digitalisierte Schwingung im Chip-RAM ab Adresse $40000 abgelegt haben, die eine Länge von 32 Bytes hat. Wie kann



ich nun dem Amiga mitteilen, wo diese Daten liegen?

Dazu dienen die Register »AUDxLC« und »AUDxLEN«. Das (x) steht für eine beliebige Zahl zwischen 0 und 3 und repräsentiert den Ausgabekanal. Die Adresse der Schwingung (Langwort) muß in das Register »AUDxLC« und deren Länge in Worten in das Register »AUDxLEN« geschrieben werden. Wichtig ist es, dabei zu beachten, daß die Adresse der Schwingung eine gerade Adresse und die Länge der Schwingung eine gerade Anzahl von Bytes ist. Letzteres ergibt sich automatisch daraus, daß die Länge sowieso in Worten (Länge von 2 Byte) angegeben werden muß (man muß in das Register »AUDxLEN« also den Wert "Schwingungslänge/2" schreiben; das hat den Vorteil, daß Schwingungen von bis zu 128 kByte Länge verwaltet werden können). Das Beschreiben der Register sieht dann in Maschinensprache so aus:

```
move.l #$40000,$dff0a0
;die Adresse der Schwingung
in AUDOLC
move.w #16,$dff0a4
;Länge der Schwingung (32 B
ytes) in AUDOLEN
```

Als nächstes müssen wir die Lautstärke des auszugebenden Tons angeben. Dazu muß in dem Register AUDxVOL ein Wert zwischen 0 und 64 eingetragen werden. Je höher dieser Wert, desto lauter klingt der Ton. Obwohl die Lautstärke eine Zahl bis höchstens 64 erreichen kann, muß sie als Wort (2 Byte) geschrieben werden, da das Register 2 Byte lang ist:

```
move.w #64,$dff0a8
;volle Lautstärke in AUDOVOL
```

Schließlich müssen wir noch die Tonhöhe angeben. Normalerweise wissen wir, daß wir einen Ton mit einer bestimmten Frequenz, zum Beispiel 440 Hz (dies entspricht dem Kammerton "a"), spielen wollen. Beim Amiga wird die Frequenz aber nicht direkt angegeben, vielmehr muß man die Anzahl der Buszyklen zwischen zwei ausgelesenen und ausgegebenen Sample-Bytes in das Register AUDxPER (Sample-Period) schreiben. Doch wie geschieht die Umrechnung von Frequenz zu »Sample-Period«? Dazu muß man wissen, aus wie vielen Samples eine Schwingung besteht, um zunächst die Anzahl der auszugebenden Samples pro Sekunde



(Sampling-Rate) zu berechnen. Dies geschieht mit folgender Formel:

Sampling-Rate = Tonfrequenz * Samples pro Schwingung

Ein Buszyklus im Amiga dauert genau 279.365 Nanosekunden. Daraus kann man nun die Sample-Period folgendermaßen berechnen:

```
Sample-Period = 1 / (Sampling-Rate * 279.365 * 10↑-9)
```

Da die Sampling-Rate, wie weiter oben bereits erwähnt, auf zirka 28 kHz beschränkt ist, sollte somit die Sample-Period nicht kleiner als 124 gewählt werden. Bei kleineren Werten kann es vorkommen, daß die Daten nicht schnell genug aus dem Speicher gelesen werden können.

Mathematik muß sein – die Daten müssen berechnet werden

Dies sieht alles sehr kompliziert aus (ist es auch), wir werden jedoch später ein Beispiel zeigen, wie man die Sample-Period in Maschinensprache effektiv und schnell berechnet. Doch in unserem Beispiel mit dem Kammerton "a" werden wir die Umrechnung vorwegnehmen, die Sample-Period beträgt 254:

```
move.w #254,$dff0a6
;ca.440 Hz Frequenz in AUD0 PER
;bei einer Schwingungslänge von 32 Bytes
```

Nun sind dem Amiga alle Parameter des Tons bekannt. Doch was hören Sie? Sie hören nichts. Wir haben nämlich vergessen, die Tonausgabe zu starten. Die Tonausgabe beim Amiga geschieht über den DMA (Direct Memory Access).

Dies ist ein Mechanismus, der in der Lage ist, selbständig und ohne Zuhilfenahme des Prozessors auf den Chip-Speicher zuzugreifen. Sämtliche Video- und Audiodaten werden vom DMA ausgelesen und an die entsprechende Peripherie weitergegeben. Der Amiga verfügt über mehrere solcher DMA-Kanäle, die quasi alle gleichzeitig Daten auslesen können. Uns interessieren hier nur die vier Audio-DMA-Kanäle. Schaltet man den DMA-Kanal, der für den Tonkanal 0 zuständig ist, ein, so ertönt der über die Register eingestellte Ton am linken Lautsprecher. Er endet erst, wenn der entsprechende DMA-Kanal wieder ausgeschaltet wird. Die DMA-Kanäle werden über das DMA-Kontrollregister »DMACON« ein- und ausgeschaltet. Dabei repräsentieren die Bits 0 bis 3 dieses Registers die Audio-DMA-Kanäle 0 bis 3. Man kann jedoch nicht so einfach durch einen Schreibzugriff gleichzeitig Bits setzen und löschen, denn das Bit 15 gibt an, ob die restlichen gesetzten Bits (im angegebenen Wort) im Register »DMACON« gesetzt oder gelöscht werden sollen. Ist Bit 15 gesetzt, so bedeutet dies, daß die anderen Bits eingeschaltet werden. Um also Kanal 0 einzuschalten, muß eine $8201, um ihn auszuschalten eine $0001 in das Register »DMACON« geschrieben werden (Bit 9 sollte beim Einschalten mitgesetzt werden, da es einen Globalschalter für alle DMA-Kanäle darstellt).

```
move.w #$8201,$dff096
;Tonausgabe starten
```

Der Amiga beginnt nun die Sample-Daten mit den eingestellten Parametern auszugeben. Er beginnt an der Adresse, die in das Register »AUDxLC« eingetragen wurde, und gibt nun in gleichmäßigen Abständen so viele Datenbytes aus, wie bei »AUDxLEN« (mal zwei) angegeben ist. Sind alle Bytes ausgegeben, fängt er wieder von vorne an, und zwar solange, bis der DMA-Kanal wieder ausgeschaltet wird:

```
move.w #$0001,$dff096
;Tonausgabe stoppen
```

Wichtig ist hierbei, daß die Tonausgabe erst dann wirklich stoppt, wenn der nächste Wert vom DMA ausgegeben werden würde. Das ist besonders dann wichtig, wenn ein neuer Ton nach dem alten gespielt werden soll; hier muß darauf geachtet werden, daß der DMA erst dann wieder eingeschaltet wird, wenn sichergestellt ist, daß seit dem Ausschalten des letzten Tones genügend Zeit verstrichen ist. Andernfalls würde der DMA das zwischenzeitliche Ausschalten gar nicht registrieren und die vorherige Schwingung weiterspielen. Je größer die Sample-Period des vorherigen Tons war, desto länger muß gewartet werden, bis der neue Ton eingeschaltet werden darf. Notfalls muß eine kleine Warteschleife aushelfen. Jedesmal, wenn der DMA beginnt, die Sample-Daten von vorne auszulesen, kopiert er sich den Inhalt des Registers »AUDxLC« in ein internes Register. Dies bedeutet, daß »AUDxLC« nach dem Starten des Tones, das heißt also nach der Ausgabe des ersten Wertes, sofort verändert werden kann, ohne daß der laufende Ton beeinflußt wird. Auch hier gilt, daß man eventuell einen Moment warten muß, um sicherzustellen, daß der DMA schon den ersten Wert ausgegeben hat. Erst wenn der DMA mit der Schwingung durch ist, registriert er die Veränderung im Register »AUDxLC«.

Diese Eigenschaft kann man ausnutzen, um digitalisierte Klänge genau einmal abzuspielen (eine Trommel sollte nur einmal gespielt werden). Dazu schreibt man nach einer kurzen Pause nach dem Einschalten der Tonausgabe in das Register »AUDxLC« die Adresse einer 2 Byte langen Schwingung mit den Sample-Daten 0 und in das Register »AUDxLEN« eine 1. Dadurch erklingt die eigentliche Schwingung nur einmal vom Anfang bis zum Ende. Immer wenn der Inhalt von »AUDxLC« in das interne Register übertragen wird, löst der Amiga einen Interrupt Level 4 aus. Für jeden Soundkanal wird ein anderer Interrupt ausgelöst. Diese Interrupts werden nur selten verwendet. Ein Beispiel dazu: Wenn die Zahl der Durchläufe der Schwingung festgelegt ist, können so die Durchläufe im Interrupt mitgezählt werden. Wenn die Schwingung nur einmal durchlaufen werden soll, so kann man, alternativ zu der oben erwähnten Methode, im Interrupt nach Beendigung der Schwingung, den Tonkanal ausschalten, während der Ton in der vorherigen Methode zwar nicht hörbar, aber dennoch weiterspielt.

Was wäre die Theorie ohne die Praxis?

Eine kleine Demonstration der hardwarenahen Tonerzeugung ist in Listing 1 aufgeführt. Geben Sie bitte dieses Listing mit dem Devpac- oder dem Seka-Assembler ein, und starten Sie das Programm. Es wird ein Dur-Akkord mit einer Sinusschwingung gespielt. Sobald Sie die linke Maustaste drücken, wird die Tonausgabe gestoppt. Da die Sample-Daten an die Routine angehängt sind, muß das gesamte Programm im Chip-RAM stehen. Falls Sie also über Fast-RAM in Ihrem Amiga verfügen, sollten Sie vor dem Starten im CLI den Befehl »run System/nofastmem« eingeben.

Sie können nun ein wenig mit diesem Programm experimentieren, verändern Sie zum Beispiel einmal die Schwingungsform. Vergessen Sie dabei nicht, die Variable »SampleLen« auf die Anzahl der verwendeten Datenworte zu setzen. Sie können ebenfalls die Kanalnummer oder die Sample-Period verändern, um die Grenzen der Tonfrequenz zu hören. Doch wie kann man nun eine oder gar mehrere Noten spielen?

Dazu zunächst ein paar Vorinformationen. Die Tonleiter besteht bekanntlich aus zwölf Halbtönen (c, c#, d, d#, e, f, f#, g, g#, a, a#, h). Nach 12 Halbtönen, das heißt nach genau einer Oktave, verdoppelt sich die Frequenz der Töne. Da sich je zwei benachbarte Halbtöne um genau den gleichen Faktor unterscheiden, muß dieser genau die zwölfte Wurzel aus 2 sein; denn wenn

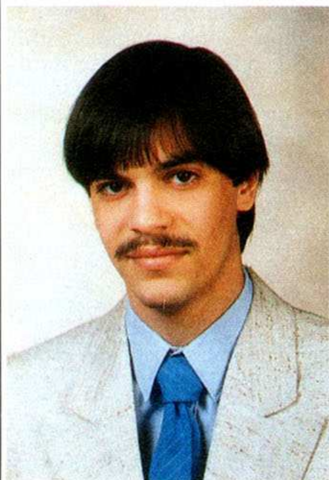
Wichtig!

Unsere kleine Exkursion in die Programmierung von Musik auf dem Amiga ist auf die Assembler-Programmierung zugeschnitten, da hierbei die Hardware am besten beschrieben werden kann. Die Programme sind als Quelltextdateien und, soweit möglich, in lauffähiger Form auf unserer Databox enthalten.

Zum Assemblieren benutzen Sie bitte den Seka-Assembler. Bei der Verwendung von anderen Assemblern müssen Sie einige Veränderungen am Quelltext vornehmen, die Sie dem Handbuch Ihres Assemblers entnehmen können.

Unser Artikel kann natürlich nicht alle Feinheiten der Soundprogrammierung behandeln, für eigene kleine Experimente und erste Hilfe für größere Projekte dürfte sich der Artikel jedoch gut eignen.

Die Autoren





Name:	Urwani
Vorname:	Zuheir
geb.:	08.02.67 in Witten
Beruf:	Informatik-Student an der Uni Dortmund; 4. Semester
Hobbys:	Computer, Musik, Sport (meistens nur sehen)

Ich habe mich, seit ich in Deutschland lebe, von Anfang an für Computer interessiert. Zunächst am Schulcomputer, später an einem Sinclair-Spectrum lernte ich das Programmieren in BASIC und in Maschinensprache. Es folgte 1986 ein Commodore 128 und schließlich 1988 der Amiga. Bereits seit 1984 wurden in mehreren Computer-Zeitschriften einige Spiele und Utilities von mir/uns veröffentlicht, jeweils für die Computer, die ich gerade besaß. Seit 1986 arbeite ich praktisch nur noch zusammen mit meinem damaligen Schul- und heutigen Studienkollegen Thomas; unser letztes Projekt war das Musikprogramm »SOUNDFACTORY« für den Amiga.

Name:	Kolbe
Vorname:	Thomas
geb.:	6. 5. 1968 in Recklinghausen
Beruf:	Informatik-Student im 4. Semester an der Uni Dortmund
Hobbys:	Computer, Indie-Musik, Fantasy-Literatur, Streß

Angefangen hat die Computerei 1982 mit dem Busch Microtronic System, welches nur in Maschinensprache programmiert werden konnte. Doch das wurde schnell zu klein, und so kam dann 1983 ein VC-20 auf den Tisch. Auf dem VC-20 lernte ich dann BASIC und 6502-Maschinensprache.

1984 mußte dann ein C-64 her, auf dem ich dann schon mit meinem Kollegen Zuheir einige Spiele und Programme für Zeitschriften programmierte. Seit 1987 bin ich nun stolzer Besitzer eines Amiga 2000, und ein Ende ist noch nicht abzusehen...

eine Frequenz zwölfmal mit dieser Zahl multipliziert wird, erhält man danach die doppelte Frequenz!

Doch zurück zur Praxis. Löschen Sie nun Teil 1 aus dem ersten Listing, und geben Sie an dessen Stelle Listing 2 ein. Dieses Programm stellt bereits den Kern eines jeden Musikprogramms dar: Es spielt eine kleine Notenfolge im Interrupt. Betrachten wir die Funktionsweise einmal etwas genauer: Die Initialisierung geschieht im wesentlichen durch "Verbiegen" des Level-3-Interrupts des Amiga auf unsere neue Interrupt-Routine. Der Level-3-Interrupt ist unter anderem für den Vertical Blanking Interrupt (VBI) zuständig, der genau 50mal in der Sekunde ausgeführt wird. Seine Regelmäßigkeit macht ihn für die Musikausgabe so interessant; viele kommerzielle Musikprogramme wie zum Beispiel der »SOUNDTRACKER« oder die »SOUNDFACTORY« verwenden ebenfalls diesen Interrupt. Nach der Initialisierung wird auf einen Druck auf die linke Maustaste gewartet, und anschließend wird die Musik wieder gestoppt.

Wenn Sie das Programm vom Assembler aus aufrufen, könnten Sie auch vor der Sprungmarke »waitMouse« den Rücksprungbefehl »rts« einfügen; die Musik läuft dann trotzdem weiter, während Sie wieder arbeiten können. Die Musik läuft quasi im Hintergrund ohne spürbaren Zeitverlust. Zum Stoppen der Musik rufen Sie dann die Routine »waitMouse« auf und drücken die linke Maustaste.

In dem Listing ist vielleicht noch die Routine »Calc_Period« interessant. Diese Routine errechnet aus einem Halbton eine Sample-Period. Dazu werden vorher die Sample-Periods einer ganzen Oktave errechnet und in einer Tabelle abgelegt. Diese Oktave sollte so tief gewählt sein, daß die Sample-Periods möglichst große Zahlen ergeben (jedoch nicht so große, daß sie nicht mehr in zwei Bytes passen). Dies hat den Sinn, daß man sie bei höheren Oktaven nur durch eine Zweierpotenz teilen und nicht multiplizieren muß. Denn beim Multiplizieren verliert ein Wert an Genauigkeit, während er bei der Division kaum an Genauigkeit einbüßt. Die Sample-Periods in dieser Tabelle sind allgemeingültig und daher noch nicht durch die Schwingungslänge geteilt, denn die Schwingungslänge kann sich von einer Schwingung zur anderen unterscheiden. Probieren Sie einmal, andere Melodien in die Tabelle »Noten« einzutragen, allerdings mit anderen Notenlängen (Tabelle »Laengen«). Die Notenlängen sind in 1/50-Sekunden anzugeben, was aus der Verwendung des VB-Interrupts resultiert.

Bisher haben wir nur sogenannte synthetisierte Klänge betrachtet. Das sind Klänge, die sehr kurz, periodisch und in der Regel von einfacher Natur sind. Eine Rechteck-, Dreieck-, Sinus- oder Sägezahnschwingung ist eine typische synthetisierte Schwingung. Kommen wir nun zu der anderen Art von Klängen, den rein digitalisierten Klängen. Diese durch einen Hardware-Digitizer (Sound-Sampler) aufgenommenen Klänge unterscheiden sich in einigen Punkten von den synthetisierten Klängen.

Von den synthetischen zu den digitalisierten Tönen



Als erstes sind sie sehr viel länger als die synthetisierten Klänge. Sie haben meist einige kBytes Länge und werden häufig nur einmal durchgespielt, das heißt, sie sind nicht periodisch, weil der gesamte Klang des Tons vom Anschlag bis zum Ausklingen digitalisiert wurde. Beim Digitalisieren wird eine bestimmte Sampling-Rate eingestellt, die dann genau wiederzuverwenden ist, wenn man den Klang in der gleichen Tonhöhe wieder abspielen möchte, in der man ihn aufgenommen hat. Sollen andere Tonhöhen erzeugt werden, so muß man die Sampling-Rate berechnen. Im Programm von Listing 3 wird genau dieser Weg eingeschlagen. Das Listing besteht aus 3 Teilen, die wie dort beschrieben eingegeben werden müssen. Das Programm leistet im Prinzip das gleiche wie das Programm in Listing 2, jedoch wird jetzt ein digitalisiertes Instrument zum Abspielen der Melodie verwendet. Sie werden sicherlich irgendwelche Instrumente, die im IFF-8SVX-Standard gespeichert sind (dies ist ein Musik-Standard auf dem Amiga), besitzen, denn viele Musikprogramme verwenden solche Instrumente (zum Beispiel Sonix oder Audiomaster). Geben Sie den Namen des Instrumentes Ihrer Wahl hinter der Marke (Label) »Instrument_Name« an, und starten Sie das Programm. Die Routine »Load_IFF« lädt dieses Instrument dann in das Chip-RAM und liefert die Parameter des Instrumentes zurück. Die Routine »Ton_Ein« wurde nun so verändert, daß sie die Schwingung nur einmal abspielt, und die Routine »Calc_Period« so, daß sie von einer gegeben Sample-Period ausgeht und diese erweitert.

Hier wurde noch ein kleiner Trick verwendet, und zwar



haben wir in der Multiplikationstabelle alle Zahlen (f↑0, f↑1, f↑2, ..., f↑11) mit f = 1/12te Wurzel aus 2 eingetragen. Diese Zahlen haben wir jedoch alle mit 32768 erweitert. Somit werden die Werte ausreichend groß, so daß die Nachkommastellen vernachlässigt werden können und nur noch mit ganzen Zahlen gerechnet werden muß. Nach der Multiplikation einer Sample-Period mit einer Zahl aus dieser Tabelle muß das Ergebnis natürlich wieder durch 32768 geteilt werden, um es wieder zu korrigieren. Doch ist dies in Maschinensprache sehr einfach, und man büßt keine Genauigkeit ein, da das Produkt eine sehr große Zahl ist, die wir durch die Division verkleinern.
Schauen Sie sich in Ruhe die Listings an, und experimentieren Sie viel damit, denn Probieren geht über Studieren, und nur so können Sie ein Gefühl im Umgang mit den Hardware-Registern des Amiga und den relativ aufwendigen Berechnungen bekommen. Sie werden beim genaueren Untersuchen der Routinen bestimmt noch das eine oder andere Interessante finden, auf das wir hier nicht näher eingegangen sind, doch halten wir diese Routinen für "durchschaubar", wenn man sich etwas näher mit ihnen beschäftigt.

(jb)

```
;------------- Listing 1 -------------------------------

; Eine Routine zur Demonstration einer Tonausgabe
; Unbedingt im Chip-RAM ausführen; notfalls vor dem
; Start:   run nofastmem    eingeben

RegBase = $dff0a0                 ;Basisadresse der Audio-Register
AUD0LC  = $0                      ;Offsets der Audioregister
AUD0LEN = $4                      ;des Tonkanals 0
AUD0PER = $6
AUD0VOL = $8
DMACON  = $dff096                 ;DMA-Kontrollregister

;------------- Teil 1 ----------------------------------

run:
        moveq #0,d0               ;Tonkanal im Register D0
        move.w #254,d1            ;Sample-Period in Register D1
        bsr Ton_Ein               ;Startet den Ton
        moveq #1,d0
        move.w #320,d1
        bsr Ton_Ein
        moveq #2,d0               ;Dur Akkord f-a-c
        move.w #213,d1
        bsr Ton_Ein
wait_mouse:
        btst #6,$bfe001           ;wartet auf linke Maustaste
        bne.s wait_mouse
        moveq #0,d0
        bsr Ton_Aus               ;Schaltet Kanal 0 aus
        moveq #1,d0
        bsr Ton_Aus               ;Schaltet Kanal 1 aus
        moveq #2,d0
        bra Ton_Aus               ;Schaltet Kanal 2 aus und beendet

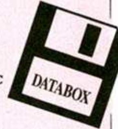
;------------- Teil 2 ----------------------------------

SampleLen = 16                    ;Länge der Schwingung in Words
Sample_Daten:
        dc.b 0,24,48,70,89,105,117,124,127,124,117,105,89,70,48,24,0
        dc.b -25,-49,-71,-90,-106,-118,-125,-127,-125,-118,-106,-90
        dc.b -71,-49,-25          ;Sinusschwingung

Ton_Ein:                          ;PARAMETER: d0=Kanalnummer
                                  ;           d1=Sample-Period
        lea RegBase,a5            ;in A5 die Registerbasisadresse
        move.l d0,d2              ;D0 nach D2 retten
        lsl.w #4,d0               ;D0:=D0*16 um Tonkanal zu adressieren
        move.l #Sample_Daten,AUD0LC(a5,d0.w)
                                  ;Adresse der Daten nach AUD0LC
        move.w #SampleLen,AUD0LEN(a5,d0.w)
                                  ;Schwingungslänge einstellen
        move.w #64,AUD0VOL(a5,d0.w)
                                  ;volle Lautstärke
        move.w d1,AUD0PER(a5,d0.w)
                                  ;Sample-Period eintragen
        move.b BitTab(PC,d2.w),d2
        or.w #$8200,d2
        move.w d2,DMACON          ;Tonausgabe starten
        rts

Ton_Aus:
        move.b BitTab(PC,d0.w),d0
        move.w d0,DMACON          ;Tonausgabe beenden
        rts

BitTab: dc.b 1,2,4,8              ;Umrechnungstabelle für 2er Potenzen
```




```
;------------- Listing 2 -------------------------------

;dieses Listing anstelle von Teil 1 des ersten Listings einfuegen !
;diese Routine spielt die Noten, die nach der Routine kommen
```

Listing: Sound-Beispiele

```

Play_Song:
        clr.w NotenPointer        ;Melodie von vorne starten
        bsr NeueNote              ;erste Note initialisieren
        move.l $6c,OldIRQ         ;alten IRQ-Vektor retten
        move.l #NewIRQ,$6c        ;IRQ-Vektor auf eigene Routine umbiegen
waitMouse:
        btst #6,$bfe001
        bne.s waitMouse           ;wartet auf linken Mausknopf
        move.l OldIRQ,$6c         ;alten IRQ-Vektor zurueckschreiben
        moveq #0,d0
        bsr Ton_Aus               ;Tonausgabe beenden
        rts

NeueNote:                         ;neue Note spielen
        moveq #0,d0
bsr Ton_aus  ;alten Ton ausschalten
        move.w #500,d0
NeueNote_Warte:                   ;kleine Warteschleife, um sicherzustellen,
        subq.b #1,d0              ;dass der DMA das Ausschalten mitbekommen
        bne.s NeueNote_Warte  ;hat
        move.w NotenPointer,d0
        lea Noten,a0
        clr.l d1
        move.b 0(a0,d0.w),d1      ;neue Note in d1
        cmp.b #-1,d1              ;Test, ob Melodie schon durch (bei -1)
        bne.s normale_Note        ;falls ja, dann Pointer auf
        clr.w NotenPointer        ;  Melodieanfang zurueckstellen
        bra.s NeueNote
Normale_Note:
        bsr Calc_Period           ;berechnet aus d1 die Sample-Period in d1
        move.w NotenPointer,d0
        lea Laengen,a1
        move.b 0(a1,d0.w),Laenge ;Laenge der gespielten Note holen
        addq.w #1,d0
        move.w d0,NotenPointer      ;Pointer weitersetzen
        clr.w d0                    ;auf Tonkanal 0
        bsr Ton_Ein                 ;Ton starten
        rts

NewIRQ:                             ;neue Interrupt-Routine
        movem.l d0-d7/a0-a6,-(sp) ;alle Register retten
        move.w $dff01e,d0
        btst #5,d0                  ;Vertical-Blanking-Interrupt?
        beq.s kein_VBI
        subq.b #1,Laenge            ;Ja, dann Zeit dekrementieren
        tst.b Laenge                ;Notenzeit abgelaufen?
        bne.s kein_VBI              ;nein, dann fertig
        bsr NeueNote                ;sonst neue Note starten
kein_VBI:
        movem.l (sp)+,d0-d7/a0-a6
        move.l OldIRQ,-(sp)         ;zur normalen IRQ-Routine
        rts

;------------- Teil 3 ----------------------------------

Calc_Period:                        ;berechnet die Sample-Period
        clr.l d0
        divu #12,d1                 ;Halbnote von Oktave trennen
        move.w d1,d0                ;in d0 ist nun die Oktave (1-6)
        clr.w d1
        swap d1                     ;und in d1 die Halbnote (0-11)
        lsl.w #1,d1
        move.w SampleTab(PC,d1.w),d1 ;in d1 nun erweiterte Sampling-Rate
        divu #SampleLen*2,d1
        and.l #$ffff,d1             ;und durch Schwingungslaenge teilen
        tst.b d0
```

Listing Sound-Beispiele




```
         beq.s Period_OK           ;Falls Oktave = 0, fert
ig
         lsr.l d0,d1               ;sonst Sample-Period Ok
tave mal halbieren
Period_Ok:
         rts

SampleTab:  dc.w 54728,51656,48757  ;Sample-Periods fuer
die 12
            dc.w 46020,43437,40999  ;Halbnoten der tiefs
ten Oktave
            dc.w 38698,36526,34476  ;(muessen noch durch
die Schwingungs-
            dc.w 32541,30715,28964  ;laenge geteilt werd
en)

;------------- Ende Teil 3 ----------------------------

Noten:      dc.b 24,26,28,29,31     ;Halbnotenfolge der
Melodie
            dc.b 29,28,26,24,28,-1  ;mit -1 abgeschlosse
n
            even

Laengen:    dc.b 24,24,24,24,12,12,12,12,24,24 ;Laengen
in 1/50 Sekunde
            even

Laenge:                  dc.w 0     ;Variable fuer die Laen
ge der akt. Note
NotenPointer:            dc.w 0     ;Zeiger auf die aktuell
e Note
OldIRQ:                  dc.l 0     ;zum Zwischenspeichern
des alten IRQs
```

```
;------------- Listing 3 -----------------------------

;---- Teil 2 aus Listing 1 nun durch folgenden Teil ers
etzen -----

Ton_Ein:                            ;PARAMETER: d0=Kanaln
ummer
                                    ;           d1=Sample
-Period
      lea      RegBase,a5           ;in A5 die Registerba
sisadresse
      move.l   d0,d2                ;D0 nach D2 retten
      lsl.w    #4,d0                ;D0:=D0*16 um Tonkana
l zu adressieren
      move.l   SampleAdr,AUD0LC(a5,d0)
                                    ;Adresse der Daten na
ch AUD0LC   (neu)
      move.w   SampleLen,AUD0LEN(a5,d0)
                                    ;Schwingungslaenge (n
eu)
      move.w   #64,AUD0VOL(a5,d0)
                                    ;volle Lautstaerke
      move.w   d1,AUD0PER(a5,d0)
                                    ;Sample-Period eintra
gen
      move.b   BitTab(PC,d2.w),d2
      or.w     #$8200,d2
      move.w   d2,DMACON            ;Tonausgabe starten
      move.w   #300,d3
Ton_Ein_Warte:                      ;Warteschleife, um si
cherzustellen, dass
      subq.w   #1,d3                ;der DMA schon min. e
inen Wert ausgegeben
      bne.s    Ton_Ein_Warte        ;hat, bevor AUD0LC ve
raendert wird
      move.l   #LeerSound,AUD0LC(a5,d0)   ;anschliessend A
UD0LC auf 2
      move.w   #1,AUD0LEN(a5,d0)    ;0 Bytes biegen, dami
t die Schwingung
      rts                           ;nicht wiederholt wir
d.

LeerSound: dc.w 0                   ;ein lautloses Instru
ment !!

Ton_Aus:
      move.b   BitTab(PC,d0.w),d0
      move.w   d0,DMACON            ;Tonausgabe beenden
      rts

BitTab:       dc.b 1,2,4,8          ;Umrechnungstabelle f
uer 2er Potenzen

SampleAdr:       dc.l 0             ;Adresse der Schwingu
ng im Speicher
SampleLen:       dc.w 0             ;Laenge der Schwingun
g in Words
SamplePeriod:  dc.w 0               ;SamplePeriod fuer Or
iginal-Tonhoehe

;--- Teil 3 aus Listing 2 durch folgenden Teil ersetzen
 ---

Calc_Period:                        ;berechnet die Sample
-Period in d1
      clr.l    d0
      divu     #12,d1               ;Halbnote von Oktave
trennen
      move.w   d1,d0                ;in d0 ist nun die Ok
tave (1-6)
      clr.w    d1
```

Listing: Sound-Beispiele

```
      swap     d1                   ;und in d1 die Halbno
te (0-11)
      clr.w    d2
      move.b   d1,d2                ;d2 = gewuenschte Hal
bnote.w
      move.w   SamplePeriod,d1      ;d1 = aktuelle Sample
-Period.w
CalcLoop1:
      lsl.w    #1,d2
      lea      MulTab(pc),a0
      move.w   0(a0,d2.w),d3        ;d3 = Faktor entsprec
hend der Halbnote
      mulu     d3,d1                ;Multiplikation
      lsl.l    #1,d1
      swap     d1                   ;d1 = d1/32768 Sample
-Period der Halbnote
      move.w   #3,d4                ;d4 = Oktave des abge
speicherten IFF-Instr.
CalcLoop2:
      cmp.b    d4,d0                ;ist gewuenschte Okta
ve korrekt?
      beq.s    CalcEnd
      bcc.s    Halbieren
      lsl.w    #1,d1                ;falls kleiner, dann
Period=Period*2
      subq.b   #1,d4
      bra.s    CalcLoop2
Halbieren:
      lsr.w    #1,d1                ;falls groesser, dann
 Period=Period/2
      addq.b   #1,d4
      bra.s    CalcLoop2
CalcEnd:
      and.l    #$0000ffff,d1
      rts

MulTab:
      dc.w 32768,30929,29193,27555,26008,24549  ;Multipli
kations-
      dc.w 23171,21870,20643,19484,18391,17359  ;tabelle

;--- folgenden Teil ganz an den Anfang einfuegen ---

;---------- laden eines IFF-8SVX-Files ----------------

Programmstart:
      lea      Instrument_Name,a0
      bsr      Load_IFF             ;Laden eines IFF-Inst
rumentes
      tst.l    d0                   ;fehlerfrei geladen?
      bne.s    Good_Load
      rts                           ;nein, dann zurueck
Good_Load:
      move.l   d0,SampleAdr         ;Adresse der Schwingu
ng abspeichern
      lsr.l    #1,d1
      move.w   d1,SampleLen         ;Laenge des Samples i
n Worten abspeichern
      move.l   #3579546,d0          ;Umrechnung Sampling-
Rate -> Sample-Period
      divu     d2,d0                ;Sample_Period=357954
6/Sampling_Rate
      move.w   d0,SamplePeriod      ;und abspeichern
      bsr      Play_Song            ;und Musik starten
      move.l   SampleAdr,a1
      clr.l    d0
      move.w   SampleLen,d0
      lsl.l    #1,d0
      bra      Unload_IFF           ;Speicher wieder frei
geben und beenden

Instrument_Name: dc.b "ram:Klavier",0  ;zum Beispiel
      even

;-----------------------------------------------------

Load_IFF: ;Diese Routine laedt ein IFF-8SVX-Instrument
an einen
          ;von ihr reservierten Speicherbereich im Chip
-RAM
          ;Wichtig: Das Instrument darf nicht gepackt s
ein und nur
          ;eine Oktave enthalten
          ;Eingabe: A0 - Zeiger auf Filenamen (mit 0 ab
geschlossen)
          ;Ausgabe: D0 - Zeiger auf Sample-Daten (0, fa
lls Fehler)
          ;         D1 - Laenge des gesamten Samples (i
n Bytes)
          ;         D2 - Sample-Rate fuer Original-Tonh
oehe
ExecBase     = 4
OpenLib      = -408
CloseLibrary = -414
AllocMem     = -198
FreeMem      = -210
Open         = -30
Close        = -36
Read         = -42
Seek         = -66
Mode_Old     = 1005

      move.l   a0,InstName          ;Zeiger auf Instrumen
tenname merken
      move.l   ExecBase,a6
      lea      DosName,a1
      jsr      OpenLib(a6)          ;dos.library oeffnen
```



Listing: Sound-Beispiele



```
    move.l  d0,DosBase          ;Basisadresse merken
    beq     Load_IFF_Ende       ;Fehler, dos.library laesst sich nicht oeffnen
    move.l  d0,a6               ;DosBase nach a6
    move.l  InstName,d1
    move.l  #Mode_Old,d2        ;Dateizugriff: lesen
    jsr     Open(a6)            ;Instrumenten-Datei oeffnen
    move.l  d0,Handle           ;Filehandle merken
    beq     Load_IFF_Fehler1    ;wenn 0, dann Fehler
    move.l  d0,d1
    move.l  #Puffer,d2
    move.l  #12,d3
    jsr     Read(a6)            ;die ersten 12 Bytes nach <Puffer> lesen
    cmp.l   #12,d0              ;wirklich12 Bytes gelesen ?
    beq.s   Load_IFF_Ok1        ;ja, dann weitermachen
Load_IFF_Fehler3:
    tst.b   VHDR_gefunden       ;falls schon Speicher im Chip-RAM reserviert
    bne     Load_IFF_Fehler5    ;wurde, muss dieser vorher freigegeben werden
    moveq   #0,d0
    bra     Load_IFF_Fehler2    ;nein, dann Fehler
Load_IFF_Ok1:
    cmp.l   #"FORM",Puffer      ;Ist die Datei ueberhaupt im IF-Format ?
    bne.s   Load_IFF_Fehler3    ;nein, dann sofort beenden
    cmp.l   #"8SVX",Puffer+8    ;Ist die Datei ueberhaupt im 8SVX-Format ?
    bne.s   Load_IFF_Fehler3    ;nein, dann sofort beenden
    clr.b   VHDR_gefunden       ;Kontroll-Block noch nicht geladen
    clr.b   BODY_gefunden       ;Sample-Daten noch nicht geladen

Load_IFF_Loop:
    tst.b   VHDR_gefunden       ;Kontroll-Block schon geladen ?
    beq.s   Load_IFF_Chunk      ;nein, dann einen Chunk (Block) laden
    tst.b   BODY_gefunden       ;Sample-Daten schon geladen ?
    bne     Load_IFF_Fertig     ;ja, dann fertig
Load_IFF_Chunk:
    move.l  Handle,d1
    move.l  #Puffer,d2
    move.l  #8,d3
    jsr     Read(a6)            ;Chunk-Header einlesen (Typ & Laenge)
    cmp.l   #8,d0               ;wirklich 8 Bytes gelesen ?
    bne.s   Load_IFF_Fehler3    ;nein, dann Fehler
    move.l  Puffer+4,d0
    addq.l  #1,d0               ;Chunk-Laenge auf naechste gerade Zahl
    bclr    #0,d0               ;aufrunden
    move.l  d0,Puffer+4
    cmp.l   #"VHDR",Puffer      ;Kontroll-Block ?
    bne     Load_IFF_M1         ;nein, dann zu Load_IFF_M1
    move.b  #1,VHDR_gefunden    ;Flag setzen
    move.l  ExecBase,a6
    clr.l   d1                  ;in d0 Speichergroesse
    jsr     AllocMem(a6)        ;Speicher fuer VHDR-Chunk anfordern
    move.l  DosBase,a6
    move.l  d0,IFF_Hilf         ;Adresse merken
    beq     Load_IFF_Fehler3    ;kein Speicher ==> Fehler
    move.l  Handle,d1
    move.l  d0,d2
    move.l  Puffer+4,d3
    jsr     Read(a6)            ;VHDR-Chunk einlesen
    cmp.l   Puffer+4,d0         ;wirklich alle Bytes gelesen ?
    beq.s   Load_IFF_Ok2        ;ja, dann zu Load_IFF_Ok2
Load_IFF_Fehler4:
    bsr     Load_IFF_FreeMem    ;nein, dann Speicher wieder freigeben
    bra     Load_IFF_Fehler3    ;und Fehler
Load_IFF_FreeMem:               ;gibt reservierten Speicher fuer Chunk
    move.l  IFF_Hilf,a1         ;wieder frei
    move.l  Puffer+4,d0
    move.l  ExecBase,a6
    jsr     FreeMem(a6)
    move.l  DosBase,a6
    rts
Load_IFF_Ok2:
    move.l  IFF_Hilf,a1
    tst.b   15(a1)              ;Instrument gepackt ?
    bne.s   Load_IFF_Fehler4    ;ja, dann Fehler
    cmp.b   #1,14(a1)           ;mehr als 1 Oktave ?
    bne.s   Load_IFF_Fehler4    ;ja, dann Fehler
    move.w  12(a1),IFF_Rate     ;Sampling-Rate merken
    move.l  0(a1),d0
    add.l   4(a1),d0            ;Laenge der Schwingung berechnen
    move.l  d0,IFF_Len          ;und merken
```

Listing: Sound-Beispiele

```
    moveq   #2,d1               ;Chip-RAM
    move.l  ExecBase,a6
    jsr     AllocMem(a6)        ;Speicher fuer Sample-Daten anfordern
    move.l  DosBase,a6
    move.l  d0,IFF_Adr          ;Adresse merken
    beq.s   Load_IFF_Fehler4    ;kein Speicher ==> Fehler
    bsr     Load_IFF_FreeMem    ;alles ok, Speicher fuer VHDR-Chunk freigeben
    bra     Load_IFF_Loop       ;und wieder zur Hauptschleife

Load_IFF_M1:
    cmp.l   #"BODY",Puffer      ;Sample-Daten ?
    bne     Load_IFF_M2         ;nein, dann zu Load_IFF_M2
    tst.b   VHDR_gefunden       ;Kontroll-Block schon geladen ?
    beq     Load_IFF_Fehler3    ;nein (IFF-Chunks in falscher Reihenfolge)
    move.b  #1,BODY_gefunden    ;Flag setzen
    move.l  Handle,d1
    move.l  IFF_Adr,d2
    move.l  IFF_Len,d3
    jsr     Read(a6)            ;Sample-Daten einlesen
    cmp.l   IFF_Len,d0          ;wirklich alle Bytes gelesen ?
    beq     Load_IFF_Loop       ;ja, dann zurueck zur Hauptschleife
Load_IFF_Fehler5:
    move.l  IFF_Adr,a1
    move.l  IFF_Len,d0
    move.l  ExecBase,a6
    jsr     FreeMem(a6)         ;nein, dann Chip-RAM wieder freigeben
    move.l  DosBase,a6
    clr.b   VHDR_gefunden
    bra     Load_IFF_Fehler3    ;und Fehler

Load_IFF_M2:                    ;unbekannter Chunk
    move.l  Handle,d1
    move.l  Puffer+4,d2
    clr.l   d3
    jsr     Seek(a6)            ;einfach ueberlesen
    bra     Load_IFF_Loop       ;und zurueck zur Hauptschleife

Load_IFF_Fertig:                ;Instrument erfolgreich geladen
    move.l  IFF_Adr,d0
    move.l  IFF_Len,d1
    clr.l   d2
    move.w  IFF_Rate,d2
Load_IFF_Fehler2:
    movem.l d0/d1/d2,-(sp)
    move.l  Handle,d1
    move.l  DosBase,a6
    jsr     Close(a6)           ;Instrumenten-Datei wieder schliessen
    movem.l (sp)+,d0/d1/d2
Load_IFF_Fehler1:
    movem.l d0/d1/d2,-(sp)
    move.l  ExecBase,a6
    move.l  DosBase,a1
    jsr     CloseLibrary(a6)    ;dos.library wieder schliessen
    movem.l (sp)+,d0/d1/d2
Load_IFF_Ende:
    rts


Unload_IFF:  ;Diese Routine gibt den durch ein vorher geladenes
             ;Instrument benutzten Speicherplatz wieder frei
             ;Eingabe: A1 - Adresse der Sample-Daten
             ;         D0 - Laenge der Schwingung in Bytes
    move.l  ExecBase,a6
    jsr     FreeMem(a6)
    rts

IFF_Adr:    dc.l 0              ;Zeiger auf Sample-Daten
IFF_Len:    dc.l 0              ;Laenge der Schwingung in Bytes
IFF_Rate:   dc.w 0              ;Sampling-Rate der Original-Tonhoehe
IFF_Hilf:   dc.l 0              ;Hilfsspeicher
Handle:     dc.l 0              ;Filehandle der geoefneten Datei
InstName:   dc.l 0              ;Zeiger auf Instrumentenname
DosBase:    dc.l 0              ;Dos-Basisadresse
DosName:    dc.b 'dos.library',0
    even
Puffer:     dc.l 0,0,0          ;12 Bytes Hilfspuffer
VHDR_gefunden:  dc.b 0          ;Flag, ob Kontroll-Block bereits geladen
BODY_gefunden:  dc.b 0          ;Flag, ob Sample-Daten bereits geladen
    even

;----- so das war's --- viel Spass beim Probieren -----
```

Listing: Sound-Beispiele

In unserer Bücherecke finden Sie regelmäßig Bücher, die sich mit dem Amiga, seiner Programmierung oder seinen Anwendungen beschäftigen. An dieser Stelle will ich auf zwei Bücher aufmerksam machen, die sehr speziell sind; es sind nämlich Datenbücher über Prozessoren und deren Peripheriebausteine. Die Anwendungsgebiete sind damit eingeschränkt, und doch können beide zu einem wichtigen Bestandteil einer Amiga-Bibliothek werden.

Mikroprozessor-Datenbuch 1 und 2

Mikroprozessor-Datenbuch 1 enthält die Anschlußbilder und Signaldiagramme von Prozessoren aller Art, darunter die komplette Motorola-Familie, angefangen beim 6800 über den 68000 bis hin zum 68030. Auch die Rockwell-Prozessoren der 65XX-Reihe sowie die Intel-Prozessoren 8080, 8085, 8086 bis 80386 (man denke an die XT- und AT-Karte, sowie an die neue 386SX-Karte) und andere sind in dem Buch enthalten. Applikationsbeispiele und Befehlslisten gehören ebenfalls zum Buchinhalt. Technische Instruktionen über Spannungen und Ströme an den Pins runden die Einzelinformation ab.

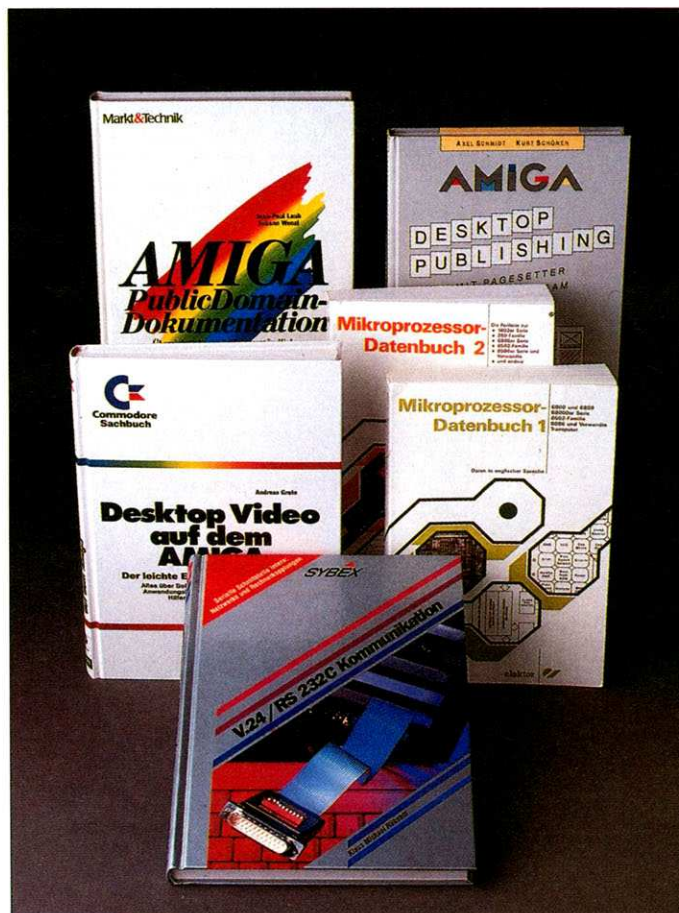
Band 2 schließlich zeigt die wichtigsten Peripheriebausteine zu den Prozessoren wie Timer, I/O-Ports, Memory-Controller und Video-Controller. Hier sind Signaldiagramme, technische Details sowie Applikationsbeispiele ebenfalls zu finden.

Die in beiden Büchern enthaltenen Daten sind ausschließlich in englischer Sprache gehalten, da sie den Original-Datenblättern entnommen wurden. Trotzdem können beide Bücher jedem Hardware-Freund oder "Selbstreparierer" empfohlen werden. Sie sind Fundgruben für jeden, der sich ein wenig mit Schaltungs-, Meß- und Regeltechnik befassen will.

(jb/vb)

Name: Mikroprozessor-Datenbuch 1 und 2
Verlag: Elektor Verlag GmbH
ISBN:3-921608-62-7
Preis: 68,– DM

Desktop-Video mit dem Amiga wird immer interessanter. Mittlerweile gibt es viele Programme, die die eigenen Videos aufpäppeln können. Mit diesem Thema befaßt sich auch dieses Buch. Nach der Einführung geht der Autor auf die Bedürfnisse der Einsteiger in dieses Metier ein. Er behandelt dabei die Frage, ob ein Modulator reicht, um Computerbilder auf Video zu überspielen, oder ob ein teures



Genlock notwendig ist. Dabei werden noch die Anschlußschemata beider Erweiterungen erläutert, so daß der Einstieg hier nicht mehr so schwer fallen dürfte.

Desktop Video auf dem Amiga

Ein Kapitel befaßt sich mit der gängigen Software. Hier werden zu einigen Programmen Einführungen gegeben. Dieses Kapitel ist nicht sehr gut gelungen, da die Einführungen einerseits zu knapp sind, um die Programme zu verstehen, und andererseits zu ausführlich, um dem Anwender einen Überblick zu geben. Es nimmt jedoch einen großen Teil des Buches ein – fast 100 Seiten. Das nächste Kapitel befaßt sich mit dem Pumuckl-Effekt, dem Programm »MovieSetter«. Dies ist sicherlich nur für die Anwender dieses Programms interessant.

Weiter geht es mit Informationen zum Digitizer »DigiView-Gold«. Hier werden Anschluß und Umgang mit diesem Gerät beschrieben. Richtig interessant wird es eigentlich erst ab dem sechsten Kapitel. Hier werden zu den verschiedenen Programmen Tips und Tricks verraten, die einige neue Möglichkeiten bieten. Beispielsweise wird hier auf den Breitwand- oder den Schlüsselloch-Effekt eingegangen. Im siebten und vorletzten Kapitel werden dem Anwender einige Anregungen gegeben, wo er seine Anlage einsetzen kann. Das Schlußkapitel beinhaltet eine Übersicht über die Programme und die Hardware. Allerdings ist diese Übersicht sehr knapp und nicht so ausführlich.

Mitgeliefert werden zwei Disketten, auf denen sich sowohl PD-Programme als auch Demos befinden. Zurück bleibt ein gemischtes Gefühl. Das Buch gibt zwar dem Anfänger einen Einblick in die Materie, allerdings kann es keine Kaufhilfe sein. Der fortgeschrittene Anwender kann auf diese "Pseudo-Kaufhilfe" verzichten und wird auch schon die meisten Kniffe kennen, die in diesem Buch beschrieben werden. Auch der Preis von 59,– DM für weniger als 190 Seiten ist etwas hoch, zumal man auf die beiden mitgelieferten Disketten verzichten könnte. Das Buch erfüllt zwar seinen Zweck, es bringt aber auch nicht mehr, was es von anderen Büchern abheben würde. Es gibt bessere Bücher.

(Andreas Polk/vb)

Name: Desktop Video auf dem Amiga
Verlag: Markt & Technik
Autor: Andreas Grote
ISBN:3-89090-312-6
Preis: 59,– DM

Der Amiga ist mit Schnittstellen und Anschlüssen gut bestückt, das weiß jeder. Zwei Ausgänge für Video (Mono- und RGB-Ausgang), ein Anschluß für weitere Disketten-Laufwerke und für den Drucker, zwei Buchsen für Audio-Anschlüsse – und eine Schnittstelle namens "seriell". Diese Schnittstelle hat eigentlich einen "Aschenputtel-Status", wird sie doch höchstens gebraucht, wenn es darum geht, die Audio-Daten zu entnehmen oder um ein Btx-Kabel oder ein Modem anzuschließen.

V.24/RS232C Kommunikation

Daß die serielle Schnittstelle des Amiga weitaus mehr zu bieten hat, als man ihr zutraut, kann man dem Buch von Klaus Michael Rübsam entnehmen. Der Autor hat dieses Buch zwar nicht speziell für den Amiga geschrieben, aber gerade darin liegt der Reiz, gibt das Werk doch auch grundlegende Hinweise auf die Verbindung mit anderen Computern.

Doch zuvor wird erst einmal der Unterschied zwischen paralleler und serieller Übertragung beschrieben. Auch serielle Schnittstellen anderer Normen wie der IEEE488/IEC 625 werden beschrieben. Für den, dem dieses Ziffernchaos nichts sagen sollte, sei erklärt, daß auch der gute alte C64 über eine ähnliche Art der Datenübertragung verfügte, in der über eine bestimmte

Geräteadresse die Peripheriegeräte seriell angeschlossen wurden. Die Anwendung der seriellen Schnittstelle und Rechnerkopplungen sowie auftretende Probleme werden ebenfalls besprochen.

Ein Kapitel ist speziell dem UART-Baustein 8250 (nicht zu verwechseln mit dem Amiga-IC 8520A) und seinen "Brüdern" gewidmet. Dieser kommt allerdings vorzugsweise in PCs zum Einsatz, da im Amiga PAULA für die serielle Datenübertragung sorgt. Ein weiteres Kapitel beschäftigt sich mit den Übertragungsparametern, der Übertragungsgeschwindigkeit und der Aufteilung von Daten in Start-, Daten-, Paritäts- und Stopbits, sowie den entsprechenden Datenprotokollen. Anschließend geht es um Handshake-Verfahren, also die Möglichkeit, Übertragungen von den beiden kommunizierenden Geräten protokollieren zu lassen. Standardisierungen und die Schnittstellen von den wichtigsten Computertypen wie Amiga, PC, Atari und Apple schließen sich an.

Sehr interessant ist das Kapitel über Rechnerkopplungen, in dem die seriellen Nullmodem-Kabel und deren Pinbelegung an den Schnittstellen verschiedener Rechner beschrieben werden. So finden Leser, die zum Beispiel Daten zwischen Amiga und PC übertragen wollen, die Anschlußbelegung von den jeweiligen Steckern, sowie direkte Hinweise zum Bau des Kabels. Ein weiteres Kapitel beschreibt die Programmierung der seriellen Schnittstelle unter GW-BASIC und Turbo Pascal – Sprachen, die leider nur auf dem PC zu finden sind. Mit etwas Wissen und Verständnis für die Funktionen lassen sich die Programmbeispiele jedoch auch auf Amiga-spezifische Sprachen umsetzen.

Den MIDI-Schnittstellen, die ja ebenfalls auf serielle Übertragung zurückgreifen, ist ebenfalls ein Kapitel des Buches gewidmet. Hier findet der technisch Interessierte wichtige Details zum Umgang mit dieser auf Musik spezialisierten Übertragung. Ausführungen zur Datenfernübertragung (DFÜ) und zu Geräten für die Fehlersuche an V24/RS232C-Schnittstellen wie Gender Changer, Jumperboxen und Loopback-Tester setzen gewissermaßen den inhaltlichen Schlußpunkt.

Ein Anhang mit Literaturhinweisen, Bezugsquellen, einem Abbildungs- und einem Tabellenverzeichnis runden das Werk ab.

Sieht man einmal davon ab, daß einige Anschlußbilder, weil falsch gezeichnet, durch Klebeschilder mit neuen Aufdrucken überklebt werden mußten, ist das Buch trotzdem empfehlenswert. Für knapp 50,– DM bekommt der Leser geballte Informationen über eine Schnittstelle, die er sonst gar nicht oder nur wenig kennen würde. Ein Buch nicht nur für Bastler, sondern auch für diejenigen, die mehr über die Hardware ihres Computers wissen wollen.

(jb/vb)

Name: V.24/RS232C Kommunikation
Verlag: Sybex Verlag Düsseldorf
Autor: Klaus Michael Rübsam
ISBN:3-88745-581-9
Preis: 49,– DM

Das Buch beschreibt auf 250 Seiten die Arbeit mit den Programmen »Pagesetter« und »Pagestream«. Leider ist der Teil über »Pagesetter« mit dem Erscheinen von »PagesetterII« nicht mehr ganz aktuell. Auf den ersten rund 50 Seiten befassen sich die Autoren mit Grundlagenwissen. Dieses Kapitel ist recht gut gelungen, nur könnten die Erklärungen ausführlicher sein. Es wird zwar gesagt, daß "weniger oft mehr" ist, wie man aber sein Dokument konkret gestaltet, wird nicht beschrieben. Hier wären einige Beispiele angebracht.

Desktop Publishing mit Pagesetter und Pagestream

Ein Kapitel befaßt sich ausschließlich mit dem Programm »Pagesetter«. Nach einer Einführung wird eine Step-by-Step-Anleitung zur Erstellung einer vierseitigen Zeitschrift gegeben. Auf den mitgelieferten Disketten befinden sich die entsprechenden Dateien, so daß der Anwender keine Zeit mit dem Abtippen von Texten verbringen muß. Abgeschlossen wird das Kapitel durch die Beschreibung einiger Utilities sowohl aus dem PD- wie auch aus dem kommerziellen Bereich und einer Liste der Shortcuts. Im großen und ganzen ist dieser Teil gut gelungen. Wer das Programm kennt, wird allerdings nicht sehr viel Neues finden. Die Beschreibungen gehen nicht wesentlich über die des Handbuches hinaus.

Der zweite Teil ist dem Programm »Pagestream« gewidmet. Auch hier werden die wichtigsten Funktionen zunächst einmal erklärt. Leider wird hier kein Beispiel angeführt, das die vielfältigen Möglichkeiten des Programms gebührend aufzeigen würde. Abgerundet wird das Buch durch ein Schriftmusterverzeichnis und eine Auflistung der DIN-Korrekturzeichen – eine sinnvolle Sache. Der Einkaufsführer listet die wichtigsten Programme auf, geht allerdings nicht auf deren Stärken und Schwächen ein. Das Buch kann allen empfohlen werden, die sich in die Materie DTP auf dem Amiga etwas einarbeiten möchten. Wer allerdings schon über das Einsteigerstadium hinaus ist, dem wird dieses Buch nicht viel bringen.

(Andreas Polk/vb)

Name: Desktop Publishing mit Pagesetter und Pagestream
Verlag: technicSupport
Autor: Axel Schmidt, Kurt Schönen
ISBN:3-926847-11-5
Preis: 69,– DM

Anfangs noch verpönt und ausgelacht, hat sich die PD-Branche mittlerweile zu einem starken Konkurrenten der kommerziellen Programme gemausert. Zwar können im Spielebereich die PD-Programme mit den kommerziellen noch nicht Schritt halten, aber dafür umso mehr im Utility- und Anwenderbereich. Hier besticht die Public Domain durch eine Vielzahl an effektiven Programmen. Wenn man dazu noch den Preis-Leistungsvergleich heranzieht, können sich die Hersteller kommerzieller Produkte bald "warm anziehen".

Das Interesse an der Public Domain zeigt sich derzeit auch bei den Neuerscheinungen auf dem Büchermarkt. Nach der Technic-Support-Serie und dem zum gleichen Thema erschienen Data-Becker-Buch, versucht sich nun auch der Markt & Technik-Verlag mit einer Dokumentation. Das Buch »Amiga Public-Domain Dokumentation« setzt sich in erster Linie mit der Fred-Fish-Serie auseinander. Dem Vorwort der Autoren und einer recht guten Einleitung, wo der Leser unter anderem mit der Legende der unterschiedlichen Zeichen vertraut gemacht wird, folgt im Buch eine Abhandlung der Fish-Disks von Nr. 1 bis Nr. 188 in numerischer Reihenfolge.

Amiga Public-Domain Dokumentation

Jedes Programm wird entsprechend als Spiele-, Anwender- oder Grafikprogramm gekennzeichnet und vorgestellt. Wenn es sich um kleine Utilities oder Grafikprogramme handelt, wird nur kurz und sachlich der Zweck des Programms erläutert. Zudem finden Sie den Programmaufruf zur Aktivierung des Programms. Etwas ausführlicher geht das Buch auf Editoren ein. Hier werden die wichtigsten Befehle erläutert und eine Aufstellung der Short-Cuts, also der Tastenkombinationen, gegeben. Dem Buch hätten einige auffrischende Bilder und Abbildungen sicherlich gut getan, so wirkt das Ganze doch etwas trocken. Zwar befindet sich in der Mitte des Buches ein Farbteil mit acht Seiten Umfang, dieser zeigt aber nicht unbedingt Wirkung auf den optischen Gesamteindruck.

Nach der großen Fred-Fish-Gala kommt auch die RW-Reihe zum Zuge, bei der die Disketten Nr. 1 bis 14 vorgestellt werden. Als Bonbon zum Abschluß wird die Diskette Nr. 1 von CCI präsentiert. Eine Auflistung in lexikalischer Reihenfolge rundet das Buch ab. Dieses Buch ist ein recht nützlicher Helfer im Umgang mit der PD und stellt übersichtlich eine Vielzahl von Programmen vor.

(Jürgen Seibel/vb)

Name: Amiga Public-Domain Dokumentation
Verlag: Markt & Technik
Autoren: Jean Paul Laub, Johann Wenzel
ISBN: 3-89090-242-1
Preis: 49,– DM

GFA für AMIGA

GFA-BASIC 3.5 Interpreter Amiga

Weiterentwicklung des GFA-BASIC 3.0 Interpreter mit 35 zusätzlichen Befehlen aus der linearen Algebra und Kombinatorik. Außerdem verbesserte Editor-Eigenschaften (Funktionen falten und Suche in Kopfzeilen gefalteter Funktionen bzw. Prozeduren) **DM 228,–** neu

GFA-BASIC 3.5 Compiler

Mit dem integrativen Compiler werden Ihre GFA-BASIC-Programme noch schneller.
Viele Optionen und Linker (kompatibel zu A-Link und B-Link) für andere Programmiersprachen im Lieferumfang enthalten. **DM 139,–** neu



Der Einstieg in GFA-BASIC 3.0 Amiga

Ein Lehrbuch für Programmieranfänger.
Dietmar Schell vermittelt auch dem unerfahrenen Programmierer Ideen und Anwendungsbeispiele für das Programmieren in GFA-BASIC. 248 Seiten, Hardcover, ISBN 3-89317-009-X **DM 29,–**



Training für Fortgeschrittene GFA-BASIC 3.0

Wer schon Erfahrung auf dem Amiga oder in irgendeinem BASIC-Dialekt hat, wird von den beiden Autoren bestens betreut.
Man erfährt und lernt eine Menge über Programmiertricks, nützliche und verwendbare Prozeduren, Anwendungen und die Besonderheiten des GFA-BASIC für Amiga. **DM 49,–**
329 Seiten, Hardcover, inkl. Diskette, ISBN 3-89317-010-3

neu

GFA-ASSEMBLER Amiga

Professioneller Makro-Assembler für 68000-Programmierer:
Leistungsfähiger Editor mit integriertem Assembler und Linker.
Nachladbarer Debugger.
Jetzt auch für die Commodore-Amiga-Computer lieferbar. **DM 149,–**

ZOETROPE

Das Computer-Animationssystem für Ihren Amiga mit der Funktionalität und den Eigenschaften, die man nur bei erstklassigen Grafiksystemen findet.
Das professionelle 2D-Animationsprogramm von ANTIC-Software, exklusiv von GFA. Umfangreiches Handbuch und Programm in Deutsch. **DM 198,–**

HÄNDLERANFRAGEN ERWÜNSCHT!
Bitte fordern Sie unser Händler-Info an!
Tel.: 0211 - 55 04-0

GFA Systemtechnik GmbH
Heerdter Sandberg 30
D-4000 Düsseldorf 11
Tel. 0211/55 04-0 Fax 0211/55 04 44



Arno Wolff

Versuche mit der Lichtmaschine

Der Druckerport mal anders genutzt

Falls Sie beim Lesen der Überschrift an ein bestimmtes Autoteil gedacht hatten – vergessen Sie es. In diesem Artikel geht es wieder um den Amiga oder besser um seinen Druckerport, den man auf einfache Weise zu einer Lichtorgel umbauen kann. Sie lachen, denken an einen Aprilscherz? Dann lesen Sie mal weiter.

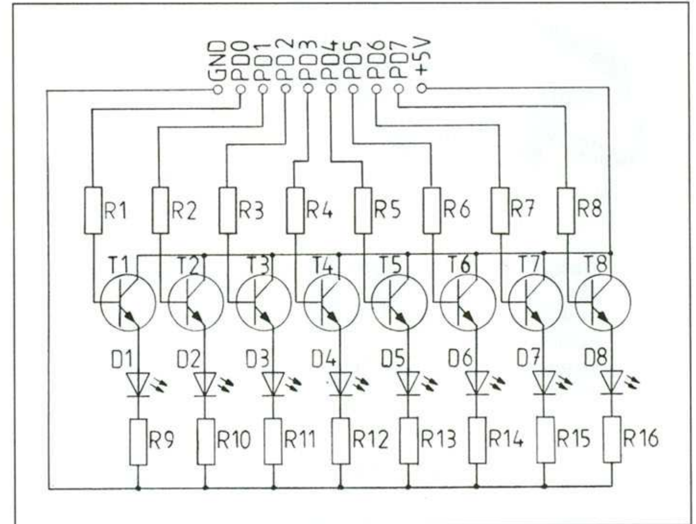


Abb. 1. Der Schaltplan unserer einfachen Lichtmaschine

Der Amiga ist, wie Sie wohl wissen, eine Supermaschine. Mit ihm kann man fast alles machen. Aber neben all den nützlichen Dingen, wie Textverarbeitung, Grafik und Musik, kann man ihn auch in den heimischen Partykeller stellen und als Lauflichtsteuercomputer benutzen.

Amiga als Beleuchter – nichts ist unmöglich!

Die beiden hier vorgestellten Interface-Schaltungen erlauben dem Amiga, eine Kette aus acht Leuchtdioden beziehungsweise acht großen 220-Volt-Lampen anzusteuern. Das Programm dazu ermöglicht das Erstellen und das Verändern eigener Sequenzen. Diese Sequenzen können auf Diskette abgespeichert, invertiert, umgedreht, mit anderen Sequenzen von der Diskette verbunden und natürlich auch ausgegeben werden. Die Programmierung erfolgt mit der Maus. Bei der Schaltung (Abb. 1) erfolgt die Ausgabe der Sequenz über acht Leuchtdioden. Jedes einzelne Leuchtmuster der Sequenz ist als Byte (8 Bit) im Speicher des Amiga abgelegt. Das Programm macht beim Ausgeben der Sequenz nichts anderes, als die einzelnen Leuchtmuster in einstellbaren Abständen an die Centronics-Schnittstelle auszugeben.

Dieses an den acht Datenleitungen der Schnittstelle anliegende Datenbyte geht über die acht Vorwiderstände "R1" bis "R8" direkt auf die zugehörigen Transistoren "T1" bis "T8". Jedes Datenbit ist für einen Transistor zuständig. Hat dieses Bit eine "1", steuert der zugehörige Transistor durch, und die zugehörige Leuchtdiode leuchtet. Ist dieses Bit eine "0", sperrt der Transistor, und die Leuchtdiode bleibt dunkel. Die Vorwiderstände "R9" bis "R16" schützen die Leuchtdioden vor zu hohem Strom und damit vor der Zerstörung.

Bei der Schaltung (Abb.2) erfolgt die Ausgabe über acht 220-Volt-Lampen. Die Datenbits gelangen über die Vorwiderstände "R1" bis "R8" auf die Optokoppler "IC1" bis "IC8". Die einzelnen Fototransistoren in den Optokopplern schalten bei Lichteinfall, also bei einer "1" eines Datenbits, positives Potential auf das Gate des zugehörigen Thyristors. Dieser schaltet dann die betreffende Lampe ein. Eine "0" als Datenbit läßt die Leuchtdiode im Optokoppler nicht leuchten. Damit sperrt der Fototransistor, und es gelangt kein positiver Impuls an das Gate des jeweiligen Thyristors. Folglich sperrt auch dieser, und die Lampe bleibt dunkel. Die Dioden "D1" bis "D6" sowie der Widerstand "R17" und der Kondensator "C1" werden für die Gleichrichtung der Netz-Wechselspannung benötigt, da der Optokoppler und die Thyristoren nur mit Gleichspannung arbeiten. Beim Aufbau der Schaltungen sollte man sehr vorsichtig vorgehen, da ein Kurzschluß Ihrem Amiga sehr schaden könnte.

Außerdem ist darauf zu achten, daß die Schaltung nach Abb. 2 mit einer Spannung von 220 Volt betrieben wird und daher absolut berührungssicher aufgebaut werden muß (zum Beispiel in ei-

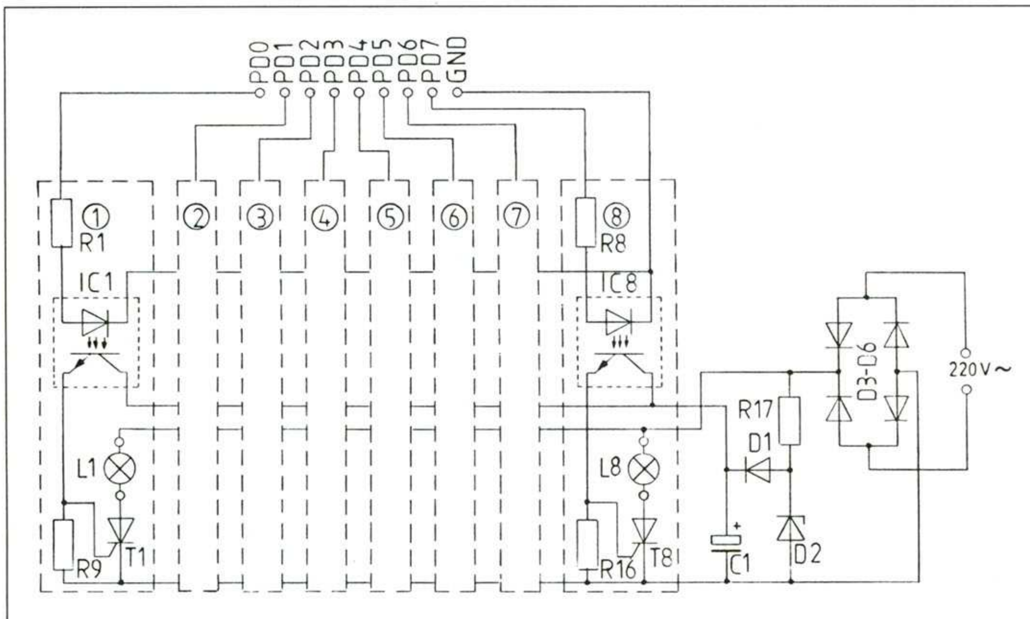


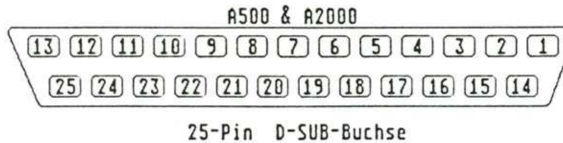
Abb. 2. Für den Partykeller gedacht. Die Profiausführung mit Lampen und Thyristoren

Wichtig!

Unsere Bauanleitung für die einfache Lichtmaschine ist ungefährlich, solange es um Menschen geht. Für den Computer allerdings ist es "überlebenswichtig", daß die Schaltung nie in eingeschaltetem Zustand an den Port gesteckt wird.

Schaltung 2 wird mit Netzspannung betrieben, hier ist unter allen Umständen auf Sicherheit zu achten. Wer sich mit der Elektrotechnik nicht auskennt, der sollte unbedingt einen Fachmann zu Rate ziehen. Spannungen von 220 Volt sind bei Berührung lebensgefährlich, deshalb Sorgfalt walten lassen!

Die Belegung der 25-poligen D-Sub-Buchse beim Amiga 500 & Amiga 2000



Pin:	Bedeutung:
1	Strobe
2	PD0 Daten-Bit 0
3	PD1 Daten-Bit 1
4	PD2 Daten-Bit 2
5	PD3 Daten-Bit 3
6	PD4 Daten-Bit 4
7	PD5 Daten-Bit 5
8	PD6 Daten-Bit 6
9	PD7 Daten-Bit 7
10	Acknowledge
11	Busy
12	Paper Out
13	Select - On Line
14	+ 5 Volt
15	nicht belegt
16	Reset
17-25	Gnd

Folgende Leitungen sind beim Amiga 1000 anders belegt. Vorsicht !

Pin:	Bedeutung:
14-22	Gnd
23	+ 5 Volt
24	Unbelegt

Abb. 3. Nochmal zur Erinnerung: Die Pinbelegung des Druckerports

nem Kunststoffgehäuse) oder bei einem Metallgehäuse der Schutzleiter unbedingt erforderlich ist. Außerdem ist darauf zu achten, daß gepolte Bauteile wie die Dioden, der Kondensator, die Thyristoren, die Optokoppler, die Transistoren und die Leuchtdioden richtig herum eingebaut werden.

Das Programm ist komplett in AmigaBASIC geschrieben und ist daher recht langsam. Diese Programmiersprache hat aber den Vorteil, daß jeder Amiga-Besitzer darauf zugreifen kann. Die Geschwindigkeit spielt beim Programmieren von Sequenzen absolut keine Rolle, und die Ausgabe der Sequenz ist unter AmigaBASIC schnell genug. Im folgenden werden die einzelnen Programmfunktionen erläutert:

Dateiname: In diesem Feld wird der Name der aktuellen Sequenz ausgegeben. Es werden nur die ersten 13 Zeichen angezeigt.

Position: Hier wird die Position innerhalb der Sequenz angezeigt, an der sich das Programm beim Programmieren oder beim Ausgeben befindet. Es sind maximal 999 Positionen möglich.

Pause + /Pause -: Durch diese Felder läßt sich die Länge der Pausen zwischen den einzelnen Positionen beim Ausgeben durch Anklicken einstellen, wobei Werte zwischen »0« und »990« in Zehnerschritten möglich sind.

Laden: Nach dem Anklicken dieser Box wird nach dem Dateinamen gefragt, unter dem die Sequenz auf der Diskette gespeichert ist. Anschließend wird die Sequenz, falls auf der Diskette im Laufwerk »Df0:« im Unterverzeichnis »Sequenzen« vorhanden, geladen. Es wird zuerst die Länge der Pausen und dann werden die eigentlichen Daten gelesen. Falls keine Sequenz geladen werden soll, muß die Frage nach dem Dateinamen mit [Return] beantwortet werden.

Speichern: Nachdem die Frage nach dem Dateinamen beantwortet ist, wird zuerst die Länge der Pausen und dann die Daten selber auf der Diskette im Laufwerk »Df0:« im Verzeichnis »Sequenzen« abgelegt. Falls die Sequenz nicht gespeichert werden soll, einfach bei der Frage nach dem Dateinamen [Return] drücken.

Disk-Inhalt: Durch Anklicken dieser Box wird der Inhalt des Verzeichnisses »Sequenzen« auf der Diskette im Laufwerk »Df0:« auf dem Bildschirm ausgegeben.

Set: Set dient zum Festlegen einer Position in der aktuellen Sequenz. Das im Fenster angezeigte Muster wird damit an der im Fenster »Position« angezeigten Position in die Sequenz übernommen. Pro Sequenz lassen sich 999 Positionen speichern.

Inv: Damit läßt sich die gesamte Sequenz invertieren, das heißt, aus einer "Null" wird eine "Eins" und umgekehrt. Die so veränderte Sequenz läßt sich dann wieder ausgeben oder abspeichern.

Back: Die Sequenz läßt sich hiermit drehen. Sie läuft praktisch rückwärts.

Pos + /Pos -: Dient zum genauen Bestimmen einer Position innerhalb einer Sequenz.

Ins: Damit ist es möglich, in die aktuelle Sequenz an mit »Pos + « und »Pos-« positionierter Stelle Daten einzufügen.

Clr: Durch Anklicken dieser Box wird die aktuelle Sequenz nach einer Sicherheitsabfrage aus dem Speicher gelöscht.

Delete: Diese Anweisung dient zum Löschen beliebiger Stellen (»Pos + /Pos-«) aus der aktuellen Sequenz.

Ausgeben: Mit dieser Anweisung wird die Ausgabe der Sequenz eingeleitet. Die einzelnen Daten (Positionen) der Sequenz werden nacheinander ausgelesen und mit einer Pause (»Pause«) ausgegeben. Durch Klicken im Sichtfenster läßt sich die Ausgabe mit verminderter Geschwindig-

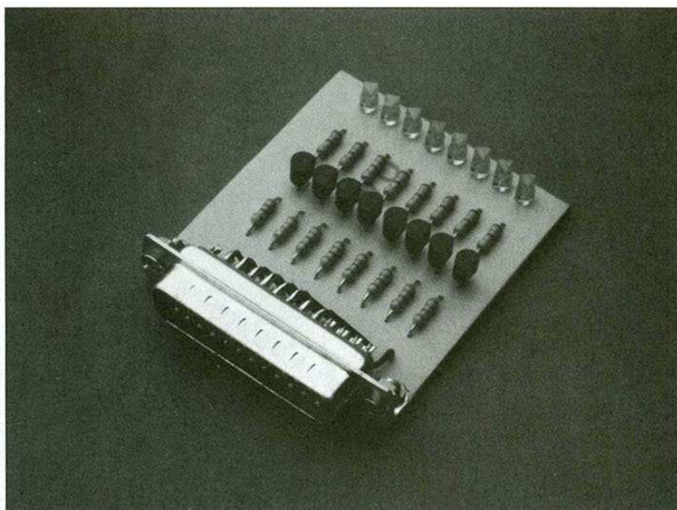


Bild 1. Die LED-Lichtmaschine kann zum Beispiel so aufgebaut werden

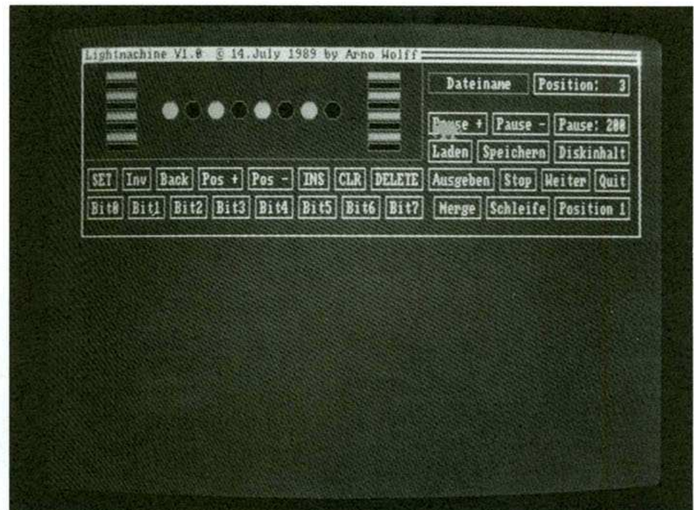


Bild 2. Das Steuerprogramm für die Lichtmaschine

keit auch auf dem Bildschirm verfolgen. Ausgeschaltet wird diese Funktion durch erneutes Klicken im Sichtfenster. Die Ausgabe an die Centronics-Schnittstelle ist immer eingeschaltet.

Stop: Mit Stop läßt sich die Ausgabe anhalten.

Weiter: Die Ausgabe der Sequenz wird an der gestoppten Position fortgesetzt.

Quit: Das Programm wird nach einer Sicherheitsabfrage beendet.

Bit0-Bit7: Diese Kästchen dienen zum Programmieren der einzelnen Positionen einer Sequenz. Durch Anklicken eines der Kästchen erscheint der dazugehörige Kreis im Sichtfenster rot und die betreffende Leuchtdiode beziehungsweise Glühlampe leuchtet. Durch erneutes Anklicken desselben Kästchens wird dies wieder rückgängig gemacht. Ist das Muster für eine Position auf diese Weise festgelegt, wird es durch Klicken im Feld »SET« in die Sequenz übernommen.

Merge: Diese Funktion läßt das Anhängen einer Sequenz im Verzeichnis »Sequenzen« auf der Diskette im Laufwerk »Df0:« an die aktuelle Sequenz zu. Nach dem Anklicken wird, wobei sich eine Sequenz im Speicher befinden muß, nach dem Dateinamen der anzuhängenden Sequenz gefragt. Ist die Datei verfügbar, wird sie geladen und an die im Speicher befindliche Sequenz angehängt.

Schleife: Mit diesem Schalter kann man zwischen einfacher und endloser Ausgabe umschalten.

Position 1: Durch Anklicken dieser Box wird die Position auf »1« und damit die Sequenz an den Anfang gesetzt.

Das Programm läuft auf allen Amigas mit mindestens 512 kByte. Bei 512 kByte muß aber das zweite Diskettenlaufwerk ausgeschaltet werden. Es bietet sich an, aus der Diskette eine Startdiskette zu machen. Damit das Programm ordnungsgemäß funktioniert, muß sich auf der Diskette das Unterverzeichnis »Sequenzen« (mit »makedir« einrichten) befinden. AmigaBASIC muß ebenfalls dort vorhanden sein. Die Schaltung arbeitet am Amiga 500 und am Amiga 2000. Für den Betrieb an einem Amiga 1000 muß die Pinbelegung angepaßt werden. Die Schaltung kann dabei unverändert bestehen bleiben.

Überlegungen zu einer guten Idee

Mit dem Programm und der kleinen Schaltung läßt sich schon einiges anfangen. Trotzdem sollten Sie, liebe Leser, sich selbst einmal Gedanken zu dem Thema machen. Wie man den Druckerport über AmigaBASIC ansteuert, zeigt Ihnen das Programm, bleibt nur die Überlegung, was man mit einer solchen Schaltung machen kann. Denkbar wären zum Beispiel eine Anti-Diebstahl-Schaltung, indem die Datenleitungen über Relais mit Lampen verbunden werden und ein Programm eine Tag- und Nachtschaltung simuliert (tagsüber aus, abends bis 11 Uhr an), oder ein automatisches Belichtungsstudio. Ihrer Kreativität sind keine Grenzen gesetzt, achten Sie aber immer auf eines: Computer-Ports sind sehr anfällig, ein falscher Anschluß – und nichts geht mehr.

Wir wünschen Ihnen auf jeden Fall viel Vergnügen und Nutzen mit Ihrer neuen Lichtmaschine.

(jb)

AMIGA DOS Toolbox

Stückliste zur Schaltung 1

R1 - R 8	= 3.3 kΩ
R9 - R16	= 180 Ω
T1 - T8	= BC 107 (BC 546)
D1 - D8	= LED

Stückliste zur Schaltung 2

R1 - R8	= 180 Ω
R9 R16	= 22 kΩ
R17	= 47 kΩ
C1	= 10 μF
D1	= 1N4001
D2	= ZD15V/400mW
D3 -	= P600J
T1 - T8	= TIC 106 D
IC1 - IC8	= IL 74
L1 - L8	= max.100W



Listings



```
SCREEN 1,640,200,4,2
WINDOW 2,"Lightmachine V1.0    14.July 1989 by Arno Wolff",(0,0)-(630,97),18,1
CLEAR ,50000&,19500
DIM sequenz(1000),backwardsequenz(1000),bit(7),testbit(7)
POKE 12575489&,255:POKE 12574977&,0:position=1:a=3:pause=200
FOR i=3 TO 10
  PALETTE i,.73,.73,.73
NEXT i
PALETTE 2,0,1,0:PALETTE 11,.8,.6,1

titelbild:
LINE (380,0)-(380,56),2
LINE (380,56)-(0,56),2
FOR i=99 TO 281 STEP 26
  CIRCLE (i,28),9,2,,,.6
  PAINT (i,28),a,2:a=a+1
NEXT i:a=3
FOR i=5 TO 47 STEP 6
  LINE (25,i)-(60,i+4),2,b
  PAINT (40,i+2),a,2
  LINE (320,i)-(355,i+4),2,b
  PAINT (340,i+2),a,2:a=a+1
NEXT i
FOR i=3 TO 339 STEP 48
  LINE (i,77)-(i+41,89),,b
NEXT i
LINE (387,5)-(499,17),2,b
FOR i=1 TO 22
  READ x1,y1,x2,y2
  LINE (x1,y1)-(x2,y2),,b
NEXT i

DATA 506,5,620,17,387,29,452,41,459,29,523,41
DATA 530,29,620,41,387,45,435,57,442,45,523,57
DATA 530,45,620,57,3,61,37,73,44,61,76,73
DATA 83,61,124,73,131,61,180,73,187,61,236,73
DATA 243,61,276,73,283,61,316,73,323,61,380,73
DATA 387,61,460,73,467,61,508,73,515,61,571,73
DATA 578,61,620,73,393,77,445,89,452,77,523,89
DATA 530,77,620,89

LOCATE 2,52:PRINT "Dateiname":LOCATE 2,65:PRINT "Position:"
LOCATE 5,50:PRINT "Pause +":LOCATE 5,59:PRINT "Pause -"
LOCATE 5,68:PRINT "Pause:":LOCATE 7,50:PRINT "Laden"
LOCATE 7,57:PRINT "Speichern":LOCATE 7,68:PRINT "Diskinhalt"
LOCATE 9,2:PRINT "SET":LOCATE 9,7:PRINT "Inv":LOCATE 9,12:PRINT "Back"
LOCATE 9,18:PRINT "Pos +":LOCATE 9,25:PRINT "Pos -"
LOCATE 9,32:PRINT "INS":LOCATE 9,37:PRINT "CLR":LOCATE 9,42:PRINT "DELETE"
LOCATE 9,50:PRINT "Ausgeben":LOCATE 9,60:PRINT "Stop"
LOCATE 9,66:PRINT "Weiter":LOCATE 9,74:PRINT "Quit"
LOCATE 11,51:PRINT "Merge":LOCATE 11,2:PRINT "Bit0"
LOCATE 11,8:PRINT "Bit1":LOCATE 11,14:PRINT "Bit2"
LOCATE 11,20:PRINT "Bit3":LOCATE 11,26:PRINT "Bit4"
LOCATE 11,32:PRINT "Bit5":LOCATE 11,38:PRINT "Bit6"
LOCATE 11,44:PRINT "Bit7":LOCATE 11,58:PRINT "Schleife"
LOCATE 11,68:PRINT "Position 1"

auswahl:
POKE 12574977&,code:LOCATE 2,75:PRINT USING "###";position
LOCATE 5,75:PRINT USING "###";pause
test=MOUSE(0)
ON MOUSE GOSUB auswertung
MOUSE ON
GOTO auswahl

auswertung:
test=MOUSE(0)
x=MOUSE(1):y=MOUSE(2)
IF y>29 AND y<41 THEN
  IF x>387 AND x<452 THEN pauseplus
  IF x>459 AND x<523 THEN pauseminus
END IF
IF y>45 AND y<57 THEN
  IF x>387 AND x<435 THEN loadflag=1:GOTO dateiname
  IF x>442 AND x<523 AND anzahl>0 THEN speicherflag=1:GOTO dateiname
  IF x>530 AND x<620 THEN diskinhalt
END IF
IF y>61 AND y<73 THEN
  IF x>3 AND x<37 THEN setzen
  IF x>44 AND x<76 AND anzahl>0 THEN invertieren
  IF x>83 AND x<124 AND anzahl>0 THEN backward
  IF x>131 AND x<180 THEN posplus
  IF x>187 AND x<236 THEN posminus
  IF x>243 AND x<276 THEN einfuegen
  IF x>283 AND x<316 AND anzahl>0 THEN loeschen
  IF x>323 AND x<380 AND anzahl>0 THEN ruecksetzen
  IF x>387 AND x<460 AND anzahl>0 THEN position2=1:GOTO ausgeben
  IF x>515 AND x<571 THEN weiter
  IF x>578 AND x<620 THEN quit
END IF
IF y>77 AND y<89 THEN
  IF posflag=1 THEN posflag=0:code=0
```



Listing: Lightmachine.BAS

```
  IF x>3 AND x<44 THEN GOTO Bitnr0
  IF x>51 AND x<92 THEN GOTO Bitnr1
  IF x>99 AND x<140 THEN GOTO Bitnr2
  IF x>147 AND x<188 THEN GOTO Bitnr3
  IF x>195 AND x<236 THEN GOTO Bitnr4
  IF x>243 AND x<284 THEN GOTO Bitnr5
  IF x>291 AND x<332 THEN GOTO Bitnr6
  IF x>339 AND x<380 THEN GOTO Bitnr7
  IF x>393 AND x<445 AND anzahl>0 AND anzahl<999 THEN mergeflag=1:GOTO dateiname
  IF x>452 AND x<523 AND anzahl>0 THEN schleife
  IF x>530 AND x<620 THEN anfangsposition
END IF
IF y>0 AND y<56 AND x>0 AND x<380 AND screenflag=0 THEN screenflag=1:GOTO auswahl
IF y>0 AND y<56 AND x>0 AND x<380 AND screenflag=1 THEN screenflag=0
GOTO auswahl

Bitnr0:
IF bit(0)=0 THEN
  PALETTE 3,1,0,0
  bit(0)=1:code=code+1
ELSE
  PALETTE 3,.73,.73,.73
  bit(0)=0:code=code-1
END IF:GOTO auswahl
Bitnr1:
  IF bit(1)=0 THEN
  PALETTE 4,1,0,0
  bit(1)=1:code=code+2
ELSE
  PALETTE 4,.73,.73,.73
  bit(1)=0:code=code-2
END IF:GOTO auswahl
Bitnr2:
IF bit(2)=0 THEN
  PALETTE 5,1,0,0
  bit(2)=1:code=code+4
ELSE
  PALETTE 5,.73,.73,.73
  bit(2)=0:code=code-4
END IF:GOTO auswahl
Bitnr3:
IF bit(3)=0 THEN
  PALETTE 6,1,0,0
  bit(3)=1:code=code+8
ELSE
  PALETTE 6,.73,.73,.73
  bit(3)=0:code=code-8
END IF:GOTO auswahl
Bitnr4:
IF bit(4)=0 THEN
  PALETTE 7,1,0,0
  bit(4)=1:code=code+16
ELSE
  PALETTE 7,.73,.73,.73
  bit(4)=0:code=code-16
END IF:GOTO auswahl
Bitnr5:
IF bit(5)=0 THEN
  PALETTE 8,1,0,0
  bit(5)=1:code=code+32
ELSE
  PALETTE 8,.73,.73,.73
  bit(5)=0:code=code-32
END IF:GOTO auswahl
Bitnr6:
IF bit(6)=0 THEN
  PALETTE 9,1,0,0
  bit(6)=1:code=code+64
ELSE
  PALETTE 9,.73,.73,.73
  bit(6)=0:code=code-64
END IF:GOTO auswahl
Bitnr7:
IF bit(7)=0 THEN
  PALETTE 10,1,0,0
  bit(7)=1:code=code+128
ELSE
  PALETTE 10,.73,.73,.73
  bit(7)=0:code=code-128
END IF:GOTO auswahl

setzen:
IF anzahl=998 THEN auswahl
IF position<anzahl+1 THEN sequenz(position3)=code:position3=position3+1:position=position+1:GOTO faerben
sequenz(position)=code
anzahl=anzahl+1:position=position+1
faerben:
IF code>0 THEN FOR i=3 TO 10:PALETTE i,.73,.73,.73:NEXT i
code=0
FOR i=0 TO 7
  bit(i)=0
NEXT i
GOTO auswahl

ruecksetzen:
IF position<anzahl+1 THEN
  FOR i=position TO anzahl
    sequenz(i)=sequenz(i+1)
  NEXT i
  sequenz(anzahl)=0:anzahl=anzahl-1:GOTO auswahl
ELSE
```

Listing: Lightmachine.BAS

```
  sequenz(anzahl)=0:anzahl=anzahl-1:position=position-
1
  GOTO auswahl
END IF

weiter:
stopflag=0
IF position3=0 THEN auswahl
position2=position3

ausgeben:
FOR i=3 TO 10
  PALETTE i,.73,.73,.73:bit(i-3)=0
NEXT i
stopflag=0
FOR i=position2 TO anzahl+1
IF i=999 THEN
  IF schleifenflag=1 THEN ausgeben
  position=999:GOTO auswahl
ELSE
  LOCATE 2,75:PRINT USING "###";i
END IF
code=sequenz(i):POKE 12574977&,code
IF screenflag=0 THEN ausgeben2
FOR j=0 TO 7
  testbit(j)=code AND 2^j
  IF testbit(j)-2^j=0 THEN PALETTE j+3,1,0,0
NEXT j
ausgeben2:
x=MOUSE(1):y=MOUSE(2):z=MOUSE(0)
IF z<>0 AND x>467 AND x<508 AND y>61 AND y<73 THEN hal
t
FOR o=1 TO pause:NEXT o
IF screenflag=0 THEN ausgeben3
FOR u=3 TO 10
  PALETTE u,.73,.73,.73
NEXT u
ausgeben3:
NEXT i
IF schleifenflag=1 THEN position2=1:GOTO ausgeben
position=anzahl+1
GOTO auswahl

halt:
stopflag=1
code=0
FOR u=3 TO 10
  PALETTE u,.73,.73,.73
NEXT u
position3=i:position=i:GOTO auswahl

schleife:
stopflag=0
IF schleifenflag=0 THEN
  schleifenflag=1:LINE (452,77)-(523,89),11,b
  GOTO auswahl
ELSE
  schleifenflag=0:LINE (452,77)-(523,89),,b
  GOTO auswahl
END IF

invertieren:
IF inversflag=0 THEN
  inversflag=1:LINE (44,61)-(76,73),11,b
  GOTO invers
ELSE
  inversflag=0:LINE (44,61)-(76,73),,b
END IF
invers:
FOR i=1 TO anzahl
  sequenz(i)=256+NOT sequenz(i)
NEXT:GOTO auswahl

backward:
IF backflag=0 THEN
  backflag=1:LINE (83,61)-(124,73),11,b
  GOTO back
ELSE
  backflag=0:LINE (83,61)-(124,73),,b
END IF
back:
FOR i=1 TO anzahl
  backwardsequenz(i)=sequenz(anzahl+1-i)
NEXT i
FOR i=1 TO anzahl
  sequenz(i)=backwardsequenz(i)
NEXT i
GOTO auswahl

einfuegen:
IF anzahl<2 OR anzahl>998 THEN auswahl
anzahl=anzahl+1:position=position+1
FOR i=anzahl TO position STEP -1
  sequenz(i)=sequenz(i-1)
NEXT i:sequenz(position)=code
GOTO faerben

pauseplus:
IF pause>980 THEN auswahl
pause=pause+10
GOTO auswahl

pauseminus:
IF pause=0 THEN auswahl
pause=pause-10
```

Listing: Lightmachine.BAS

```
GOTO auswahl

posplus:
IF position=999 OR position>anzahl THEN auswahl
posflag=1
position=position+1:code=sequenz(position)
GOTO auswahl

posminus:
IF position=1 THEN auswahl
posflag=1
position=position-1:code=sequenz(position)
GOTO auswahl

anfangsposition:
posflag=1
position=1:code=sequenz(position)
GOTO auswahl

loeschen:
WINDOW 3,"  Sequenz wirklich loeschen ?  ",(387,0)-(63
0,25),2,1
LOCATE 2,3:PRINT "(`J` fuer JA / `N` fuer NEIN)"
GOSUB taste
IF taste$="n" THEN WINDOW CLOSE 3:GOTO auswahl
anzahl=0:position=1:LOCATE 2,3:PRINT "     Einen Moment
 bitte !   "
FOR i=1 TO 999
  sequenz(i)=0
NEXT i
WINDOW CLOSE 3:LOCATE 2,50:PRINT "  Dateiname  ":GOTO
 auswahl

dateiname:
WINDOW 3,"Bitte Dateinamen eingeben !",(387,0)-(630,25
),2,1
LOCATE 2,2:LINE INPUT"Dateiname: ";seqname$
IF seqname$="" THEN WINDOW CLOSE 3:speicherflag=0:load
flag=0:mergeflag=0:GOTO auswahl
WINDOW CLOSE 3
LOCATE 2,50:PRINT "               ":LOCATE 2,50:PRINT LE
FT$(seqname$,13)
IF loadflag=1 THEN GOSUB laden:loadflag=0:GOTO auswahl
IF speicherflag=1 THEN speichern
IF mergeflag=1 THEN GOSUB laden:mergeflag=0:GOTO auswa
hl
GOTO auswahl

speichern:
OPEN "sequenzen/"+seqname$ FOR OUTPUT AS #1
PRINT #1,pause
FOR i=1 TO anzahl
  PRINT #1,sequenz(i)
NEXT i
CLOSE #1:speicherflag=0
KILL "Sequenzen/"+seqname$+".info"
GOTO auswahl

laden:
OPEN "Sequenzen/"+seqname$ FOR APPEND AS #1
IF LOF(1)=0 THEN
  CLOSE #1:BEEP:BEEP:BEEP
  LOCATE 2,50:PRINT "  Dateiname":seqname$=""
  RETURN
ELSE
  CLOSE #1
END IF
IF mergeflag=1 THEN
  i=anzahl+1
ELSE
  FOR i=1 TO 999:sequenz(i)=0:NEXT i:i=1
END IF
OPEN "Sequenzen/"+seqname$ FOR INPUT AS #1
INPUT #1,pause
WHILE NOT EOF(1)
  IF i=999 THEN CLOSE #1:anzahl=i-1:position=i:RETURN
  INPUT #1,sequenz(i):i=i+1
WEND
CLOSE #1:anzahl=i-1:position=i
RETURN

quit:
WINDOW 3,"  Programm wirklich beenden ?  ",(387,0)-(63
0,25),2,1
LOCATE 2,3:PRINT "(`J` fuer JA / `N` fuer NEIN)"
GOSUB taste
IF taste$="j" THEN SYSTEM
WINDOW CLOSE 3:GOTO auswahl

taste:
taste$=INKEY$
IF taste$="j" OR taste$="n" THEN RETURN
GOTO taste

diskinhalt:
WINDOW 3,"Disketteninhaltsverzeichnis",(410,0)-(630,18
5),2,1
FILES "Sequenzen"
mausklick:
test=MOUSE(0)
ON MOUSE GOSUB fertig
MOUSE ON
GOTO mausklick
fertig:
WINDOW CLOSE 3:GOTO auswahl
```



Listing: Lightmachine.BAS

Wichtiger Hinweis:

Da manche Listings, Dateien und Listen zu sehr in die Breite gehen, sind wir manchmal gezwungen, Zeileneinrückungen fallenzulassen, um mit der Breite einer Textspalte nicht in Konflikt zu kommen. Damit Sie beim Eingeben keinerlei Probleme haben, werden alle Dateien mit Zeilennummern versehen. Diese dienen nur der besseren Übersicht und sollten nicht mit übernommen werden. Bei der Eingabe der Dateien halten Sie sich bitte an die entsprechenden Vorgaben für die jeweiligen Programme. Mountlists sollten so eingegeben werden, wie sie im DEVS-Verzeichnis stehen. (Red.)

Gewußt wie

Die Seiten für Einsteiger und Fortgeschrittene

Auf in eine neue Runde mit Tips und Tricks rund um den Amiga. Ob Shell, CLI, BASIC, Assembler, C, Workbench – alles, wozu Ihnen ein guter Tip einfällt, können Sie hier Ihren Leserkollegen mitteilen – und eine Databox bekommen.

Weg mit dem Befehl

Text-Files, die im »Notepad« abgespeichert werden, können im CLI nicht oder nicht mehr gelesen werden, weil sie gleich nach ihrem Erscheinen aus dem Screen laufen. Schuld daran ist der am Ende der abgespeicherten Files gesetzte Befehl »CONTROL L« (↑L) (refresh the screen). Wenn mit dem Editor »MEmacs« der Befehl »↑L« am Ende der Textfiles gelöscht wird, können diese ohne »Notepad« mit »Type <Dateiname>« auch im CLI-Fenster gelesen werden, was beim Arbeiten mit der Shell oft bequem und zeitsparend ist.

(Josef Maier)

Bewegliche Icons

Wie vielleicht schon bekannt, ist es möglich, Icons fest zu positionieren, indem man etwa das »Default tool«, »Tool Types« oder ein »Comment« einträgt und abspeichert. Nun habe ich es aber lieber, wenn AmigaDOS jedesmal neu überlegt, wie die Icons wohl am besten stehen – unter anderem, weil das Chaos so immer noch kleiner bleibt, als wenn hin und wieder einzelne Icons oder Dateien gelöscht werden und der Rest nicht nachrückt. Hilfe bietet hier zum Beispiel das PD-Programm »NoIconPos«. Wer darüber nicht verfügt, kann sich auch mit einem beliebigen Dateimonitor helfen, indem er in der Icon-Datei (jene Datei, die außer dem in der Workbench sichtbaren Namen die Endkennung ».info« trägt) ab Position 58 = $3A die Byte-Folge 128,0,0,0,128,0,0,0 beziehungsweise 80,00,00, 00,80,00,00,00 einträgt.

Icon-Typen

Schon »IconEd« weist darauf hin, daß man Icons mit verschiedenen Typen tunlichst auseinanderhalten sollte. Was aber, wenn man dies irgendwann nicht kann und den Typ wechseln will? Ich habe mir zum Beispiel eine Icon-Sammlung angelegt. Und ein Icon vom Typ "Disk" läßt sich eben nur bestaunen, wenn es mit dem Namen »Disk.info« im Root-Directory einer Diskette steht! Hilfe verspricht wieder der Dateimonitor; in Position 48 = $30 steht das Byte, das den Typ des Icons bestimmt. Vorgesehen sind: 1=Disk, 2=Drawer, 3=Tool, 4=Project, 5=Garbage, 6=Device (unsinnig, da keine Informationen verfügbar) und 7=Kick (wie 6).

(Arnold Schommer)

Mausposition in Assembler

Ich habe mir ein kleines Programm in Assembler erstellt, das die Mausposition auswertet. Mit dieser Routine kann man zum Beispiel in einem Programm den Mauszeiger oder ein Sprite bewegen:

```
;*** Auf dem DEVPAC II erstellt. ***
R_Taste EQU 6
Port1 EQU $DFF00A
CIA EQU $BFE001

;*** Mausabfragen HOCH und RUNTER. ***

move.b Port1,AlteMausPos ;Mauswerte in Platzhalter
eintragen.
Wait bsr.s Maus ;Springe zur Unterroutine
btst #R_Taste,CIA ;Warte auf rechte MausTaste.
bne Wait ;Nein, dann springe wieder auf WAIT.
rts ;Programm beenden.

;*** Und hier ist die Unterroutine fuer die Maus. ***

Maus move.b AlteMausPos(pc),d0 ;Alter Wert in d0
move.b Port1,d1 ;Neuer Wert in d1
sub.b d1,d0 ;Beide voneinander abziehen.
cmp.b #10,d0 ;Gibt es einen Unterschied von 10?
bpl.s Oben ;Dann springe zu OBEN.
cmp.b #-10,d0 ;Gibt es einen Unterschied von -10?
bmi.s Unten ;Dann springe zu UNTEN.
rts

;** UNTEN und OBEN veraendern die Hintergrundfarben **

Unten
move.b Port1,AlteMausPos ;Gleich wieder neuen Wert laden.
move.w #$0900,$dff180 ;Hintergrund ROT anzeigen.
rts ;Und wieder zurueck.

Oben
move.b Port1,AlteMausPos ;Gleich wieder neuen Wert laden.
move.w #$0fff,$dff180 ;Hintergrund WEISS anzeigen.
rts ;Und wieder zurueck.

AlteMausPos
dc.b 0 ;Platzhalter fuer die Mauswerte.
```

(Christoph Olf/jb)

Vom Rollen eines Strings

Das Programm »ROLL«, das mit AmigaBASIC geschrieben wurde, soll zwei Dinge für BASIC-Neulinge demonstrieren. Zum einen, wie man mit den String-Befehlen des BASIC arbeitet, zum anderen zeigt es, wie man Text innerhalb eines Windows zentriert ausgibt. Die nachfolgende Step-by-Step-Beschreibung erläutert, wie das Programm arbeitet. Es sollte eigentlich keine Probleme beim Übernehmen der Routine in eigene Programme geben!

(Markus Zeindlinger)

Die Bedeutung der einzelnen Programmzeilen

- 1 Bildschirm löschen: »bs$« global definieren
- 2 a$ wird der Text zugewiesen, der gerollt werden soll.
- 3 »ROLL« wird aufgerufen: »ZEIT« wird für Warte-Routine benötigt: »WARTE« wird aufgerufen.
- 4 Die Leerzeichen haben die gleiche Länge, wie der erste Text, damit dieser wieder ganz verschwindet.
- 5 »ROLL« wird aufgerufen
- 6–9 Siehe 2–4
- 10 Schluß. Aus. Basta. Punkt.
- 11 »ROLL« beginnt, »a$« wird als »TEXT$« übernommen.

12 Die Länge des Strings wird in der Variable »LANG« abgespeichert.
13 Die Schleifenlänge entspricht der Zeichenlänge des Textes.
14 Der Text wird in einzelne Buchstaben zerlegt, die in der Feldvariablen »bs$« nacheinander abgelegt werden.
15 Schleifenende
16 Wartezeit = 60: Die Höhe des Fensters (in Pixel) wird durch acht geteilt (Zeilen). Die Zeilen werden durch zwei dividiert, damit der Text vertikal zentriert wird. Mit » + 1« geht's besser!
17 Die Breite des Fensters wird durch acht geteilt (Spalten). Von diesem Wert wird die String-Länge abgezogen und das Ergebnis durch zwei dividiert. So erhält man den linken Rand. » + 2« ist reine Geschmackssache! Für den rechten Rand zählt man einfach die String-Länge zum linken dazu. 18 Schleifenbeginn. Die String-Länge wird durch zwei dividiert (eine Hälfte rollt von links, die andere von rechts). » + 1« ist unbedingt notwendig, damit auch ein String mit ungerader Zeichenzahl komplett geschrieben wird!
19 Positionierung: Schreiben des linken Zeichens: Eine Spalte weiter
20 Positionierung: Schreiben des rechten Zeichens. Von der String-Länge wird der Schleifenzähler abgezogen.
21 Eine Spalte nach links für die rechten Zeichen
22 Aufrufen der Warte-Routine
23 Schleifenende
24 Ende der Routine »ROLL«
25 Warteroutine. Die Schleifendurchläufe werden als »a« übernommen.
26 Warteschleife
27 Ende der Routine »WARTE«

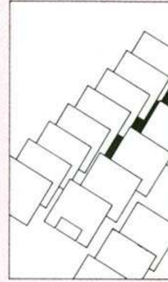
Hier das Programm:

```
1 'ROLL-Created and programmed by Mark Zeindlinger
2 CLS:DIM SHARED bs$(80)
3 a$="ROLL!!! Created and written by Mark Zeindlinger."
4 CALL Roll(a$):zeit=3000:CALL Warte(zeit)
5 a$=SPACE$(LEN(a$))
6 CALL Roll(a$)
7 a$="Ist das nix, für Eure Tips&Tricks!?!?!?!"
8 CALL Roll(a$):CALL Warte(zeit)
9 a$=SPACE$(LEN(a$)):CALL Roll(a$)
10 END
11 SUB Roll (Text$) STATIC
12 lang=LEN(Text$)
13 FOR Zaehler=1 TO lang
14 bs$(Zaehler)=MID$(Text$,Zaehler,1)
15 NEXT Zaehler
16 t=60:x=((WINDOW(3)/8)\2)+1
17 y=(((WINDOW(2)/8)-lang)\2)+2:z=y+(lang-1)
18 FOR effekt=1 TO (lang\2)+1
19 LOCATE x,y:PRINT bs$(effekt):y=y+1
20 LOCATE x,z:PRINT bs$((lang-effekt)+1)
21 z=z-1
22 CALL Warte(t)
23 NEXT effekt
24 END SUB
25 SUB Warte(a) STATIC
26 FOR b=1 TO a:
27 NEXT b
28 END SUB
```

Indische Workbench

Besonders reizvoll ist es natürlich, die Startup-Sequence den individuellen Bedürfnissen anzupassen. Meist endet dabei der Befehl »LoadWb« mit einer Reise nach Indien! Viele Anfänger wissen es nicht, weil es selten erwähnt wird: Die Workbench benötigt unbedingt die »icon.library«. Sie finden diese im Verzeichnis »libs« der Workbench-Diskette (wo auch sonst?). Legen Sie einfach auf Ihrer Startdiskette mit dem Befehl »makedir« ein entsprechendes Verzeichnis an, und kopieren Sie die »icon.library« dort hinein. Der Guru sollte nun eigentlich in seinem Tempel bleiben.

(Markus Zeindlinger)



Tausche Tip gegen DATABOX

Wir machen Ihnen wieder unser Angebot: Sie schicken einen oder mehrere kurze Tips an uns, und wenn uns diese gefallen, bekommen Sie von uns im Austausch die DATABOX des Heftes zugeschickt, in dem Ihr Tip erscheint.

Dazu gibt es jetzt noch etwas mehr:

Die drei besten Tips werden mit je 100, – DM belohnt.

Die Redaktion der AMIGA DOS sucht ab sofort unter allen Einsendern die drei **Supertips des Monats** aus, die außer der DATABOX noch je einhundert deutsche Märker (100, – DM) bekommen.
Dabei sollten Sie noch folgendes beachten:

– Schreiben Sie Ihren Tip als ASCII-Datei auf Ihre Diskette, möglichst ohne eigensinnige Verschnörkelungen, da alle Texte zur weiteren Verarbeitung auf Personalcomputer transferiert werden müssen. Sonder- und Steuerzeichen haben dabei meist seltsame Auswirkungen.

– Schreiben Sie Ihre lückenlose Anschrift sowie einen kleinen Hinweis, was auf der Diskette zu finden ist, auf den Brief und auf einen kleinen Aufkleber auf der Diskette.

Dann können Sie sicher sein, daß die DATABOX Sie erreicht. Schicken Sie das Ganze an den

DMV-Verlag
Redaktion AMIGA DOS
Kennwort "Tausche Tip gegen DATABOX"
Postfach 250
D-3440 Eschwege

Joystick in Basic

Ein Action-Spiel in BASIC zu programmieren, davon träumt so mancher AMIGA-Besitzer. Aber wie? Zuerst einmal brauchen wir eine vernünftige Joystick-Abfrage:

```
REM Joystick-Abfrage
OPEN "EXTRASD: BASICDEMOS/BALL" FOR INPUT AS 1
OBJECT.SHAPE 1, INPUT$(LOF(1),1)
CLOSE 1
x=10:y=10
OBJECT.ON 1
WHILE 1
OBJECT.X 1,x
OBJECT.Y 1,y
x=x+STICK(2)
y=y+STICK(3)
IF STICK (=) = -1 THEN BEEP
WEND
```

Dieses kurze Programm zeigt die Joysticksteuerung eines Objekts, dabei wird auch der Feuerknopf abgefragt; drückt man ihn, ertönt ein Piepston. Durch Einbau dieser Routine in eigene Programme kann man eine Joystick-Abfrage realisieren.

(Axel Terhorst/jb)

Bei der Wahl der besten Tips fiel die Entscheidung diesen Monat nur auf zwei Kandidaten. Nun, wer will sich schnell 100, – DM verdienen? Wir warten auf die Tips des Monats. Machen Sie mit, vielleicht steht Ihr Beitrag bald an dieser Stelle. Doch nun zu den Tips:

Verbesserte Shell-Befehle

Die Shell-Befehle »dir« und »list« haben beide ihre Vor- und Nachteile. Das Angenehme an »dir« ist, daß die Unterverzeichnisse getrennt von den anderen Dateien aufgelistet werden. Leider bekommt man von »dir« keine Informationen über die Dateigröße oder ähnliches; »list« wiederum liefert diese, aber dafür bleibt die Übersicht auf der Strecke.
Abhilfe schafft eine kleine Batch-Datei, die zunächst die Unterverzeichnisse ohne weitere Informationen ausgibt und dann zu allen anderen Dateien je nach Belieben Größe, Status und Datum anzeigt. Der folgende Vierzeiler zeigt zu den Dateien nur Größe und Status an. Das läßt sich leicht erweitern, indem man die Parameter bei den List-Befehlen ändert. Hier die Batch-Datei:

```
.key verz
.default verz #?
list <verz> quick nohead dirs
list <verz> nodates nohead files
```

Dieses Befehls-File sollte im Directory »C:« abgespeichert werden – am besten mit dem Namen »dir« (allerdings vorher den richtigen Befehl in »ddir« oder ähnliches umbenennen). Dann setzt man mit »protect« das Script-Bit. Verfügt man nicht über eine Festplatte, kann man die Datei auch in der Startup-Sequence ins RAM kopieren (Pfad setzen). Es ist weiter zu empfehlen, »list« resident zu machen.
Das Programm hat einen kleinen Schönheitsfehler: Gibt man als Parameter eine Datei ein, die nicht existiert, erscheint zusätzlich zur Meldung »Can't examine« noch die Anzeige »list failed returncode 20«. Diesen Fehler zu beheben, wäre leicht, nur würde das File dadurch etwas länger, und bei Batch-Dateien ist die Zeit ja sowieso immer knapp.

(Edmund Pohl / jb)

File-Requester für BASIC-Programmierer

"Einen ARP-File-Requester in AmigaBASIC programmieren – wie soll das möglich sein?" fragen sich viele BASIC-Anwender angesichts der hervorragenden File-Requester, die es unter der ARP-Library gibt. Nun, zuerst besorgt man sich die ARP-Library und installiert diese im Verzeichnis »Libs«. In dieser Library ist bereits ein File-Requester installiert, wir müssen nur noch (bescheidene Systemkenntnisse vorausgesetzt) dem BASIC den Aufruf davon in einer Datei »bmap« klarmachen. Danach heißt es nur noch "Was C- und Modula-Programmierer können, das können wir in BASIC ebensogut", nämlich eine File-Requester-Struktur aufbauen (wozu gibt es sonst Poke und Co.?). Wenn wir das alles erledigt haben, steht uns mit wenigen Programmzeilen ein schönes Werkzeug zur Verfügung. Wenn dieses Programm in eigene "Produktionen" übernommen werden soll, dann vergessen Sie bitte nicht, die »arp.bmap« ins Verzeichnis »Libs« oder ins aktuelle Verzeichnis (dort, wo das Programm laufen soll) zu kopieren. Nun das Programm, voila...

```
1 '--- Klaus Küchler --- April 90 --
2 DECLARE FUNCTION FileRequester&() LIBRARY
3 LIBRARY "arp.library"
4 DIM SDir$(10),SFile$(10)
5
6 Var: FRS$ = "" ' FileRequesterStruct
7 Titel$ = "" ' RequesterTitle
8 File$ = "" ' DefaultFile
9 Dir$ = "" ' DefaultPath
10
11 i% = 0
12 Main:
13 CLS:RESTORE
14 LOCATE 20,1
15 PRINT "Nur Mut, die Operationen werden nur simuliert!"
16 READ Titel$
17 WHILE Titel$<>"ENDE"
18 IF File$="" THEN
19 File$=CHR$(0)+"<39 Space>"
20 ELSE
21 File$=File$+CHR$(0)+"<33 SPACE>"
22 END IF
23 IF Dir$="" THEN
24 Dir$="df0:"+CHR$(0)+"<33 SPACE>"
25 ELSE
26 Dir$=Dir$+CHR$(0)+"<35 SPACE>"
27 END IF
28 Titel$=Titel$+CHR$(0) ' C-Konventionen
29 GOSUB FileRequester
30 SDir$(i%) = Dir$
31 SFile$(i%) = File$
32 i% = i% + 1
33 READ Titel$
34 WEND
35
36 CLS:RESTORE:PRINT
37 PRINT "Sie hatten folgende Auswahl getroffen:"
38 PRINT:PRINT
39 i% = 0
40 READ Titel$
41 WHILE Titel$<>"ENDE"
42 PRINT " Bei ";Titel$
43 PRINT " das Verzeichnis "+CHR$(34)+SDir$(i%)+CHR$(34);
44 PRINT " und den Dateinamen"+CHR$(34)+SFile$(i%)+CHR$(34)
45 PRINT
46 i% = i% + 1
47 READ Titel$
48 WEND
49 LIBRARY CLOSE
50 PRINT
51 PRINT "Na, ist das was?":PRINT:PRINT
52 PRINT "MERKE: Was man vom Requester zurueckbekommt,ist
   lediglich"
53 PRINT " der Name des gewählten Verzeichnisses und der Datei-"
54 PRINT " name. Was man damit macht, hat man selbst zu ver-"
55 PRINT " antworten!"
56 END
57
58 DATA "Datei LADEN "
59 DATA "Datei SPEICHERN "
60 DATA "Datei LOESCHEN "
61 DATA "ENDE"
62
63 FileRequester:
64 WHILE LEN(FRS$)<30
65 FRS$=FRS$+CHR$(0)
66 WEND , 67 adr&=SADD(FRS$):adr&=2*CLNG(adr&/2)
68 Titel& = SADD(Titel$)
69 File& = SADD(File$)
70 Dir& = SADD(Dir$)
71
72 '-- ein C-Programmierer beginnt hier mit STRUCT...
73
74 POKEL adr& , Titel& ' Titel im FileRequester
75 POKEL adr& + 4, File& ' Default File
76 POKEL adr& + 8, Dir& ' Default Verzeichnis
77 POKEL adr& +12, 0 ' Zeiger auf Windowstructur, wenn andere als
   aktuelle
78 POKE adr& +16, 96 ' FlagSet, siehe unten
79 POKE adr& +17, 0 ' Reservierte Byte fuer spätere
80 POKEL adr& +18, 0 ' Erweiterungen, so vermute ich
81 POKEL adr& +22, 0 ' wenigstens!?
82
83 '-- Funktion aufrufen und staunen !!!!!!!! --
84
85 ok& = FileRequester&(adr&)
86 IF ok& <> 0 THEN
87 ld% = INSTR(Dir$,CHR$(0)) ' C-String -> BASIC-String
88 lf% = INSTR(File$,CHR$(0))
89 Dir$ = LEFT$(Dir$,ld%-1)
90 File$ = LEFT$(File$,lf%-1)
91 ELSE
92 Dir$ = ""
93 File$ = ""
94 END IF
95 RETURN
```

Einige Hinweise noch zum »FlagSet«, soweit mir bekannt:

FlagSet Bit	Wert	Bezeichnung	Kommentar
0	1	listFunc	?
1	2	gEventFunc	?
2	4	addGadFunc	?
3	8	newWinFunc	muß gesetzt werden, wenn eigenes Window für File-Requester aufgebaut wird
4	16	newIDCMP	wie »newWinFunc«
6	32	doColor	wenn 0, dann weiß, sonst Workbenchfb.
7	64	doMsgFunc	muß gesetzt sein
8	128	doWildFunc	?

Eine neue Window-Struktur erhält man in BASIC mit WINDOW(7), aber bitte nur dann in die File-Requester-Struktur einsetzen, wenn auch tatsächlich ein Window für den File-Requester aufgebaut wird, andernfalls kommt Besuch aus Indien! Vielleicht kann mir ja jemand mehr dazu sagen?

(Klaus Küchler)

D. Kuntz

Die Sache mit den Namen

»RenDev« – Ein Device wird umgetauft

Das Multitasking-Betriebssystem des Amiga verwendet zum Ansprechen der Hardware sogenannte Devices. Manchmal wäre es ganz schön, wenn man diese einfach umbenennen könnte. Der Vorteil liegt auf der Hand: Man hat mehr Möglichkeiten, diese Devices dann auch wieder anzusprechen.

Alle Devices bilden über das DOS und den entsprechenden Handler die Verbindung zwischen Software und Ausgabegerät. So ist zum Beispiel für die Diskettenlaufwerke (DF0: bis DF3:) das Trackdisk-Device zuständig. Man unterscheidet zwischen logischen Devices und solchen, die "gemountet" werden müssen.

Logische Devices befinden sich im ROM und sind somit schon beim Booten vorhanden. Zu den logischen Devices gehören DF0: bis DF3:, CON:, PAR:, PRT:, SER:, RAW: und RAM:. Alle anderen Devices müssen zuerst in das System eingebunden werden. Das erfolgt mit dem Mount-Befehl, wobei die entsprechenden Parameter in der Mount-List angegeben werden. Hierzu gehören zum Beispiel RAD:, NEWCON: oder PIPE:. Um nun auf die Daten der Devices im System zugreifen zu können, muß man sich durch einige Strukturen hangeln, bis man ans Ziel gelangt. Ausgangspunkt ist die DOS-Library. Beim Öffnen der DOS-Library wird ein Zeiger auf auf die Library zurückgegeben. Da diese jedoch bereits vom System gebraucht und geöffnet wurde, steht dieser Zeiger schon in DOS-Base bereit. Er zeigt auf folgende Struktur:

```
struct DosLibrary
 {
  struct  Library dl_lib;
  APTR    dl_Root;
  APTR    dl_GV;
  LONG    dl_A2;
  LONG    dl_A5;
  LONG    dl_A6;
 }
```

Von hier aus wird der Zeiger »dl_Root« weiterverfolgt, der auf eine Root-Node-Struktur zeigt:

```
struct RootNode
 {
  BPTR    rn_TaskArray;
  BPTR    rn_ConsoleSegment;
  struct  DateStamp rn_Time;
  LONG    rn_RestartSeg;
  BPTR    rn_Info;
  BPTR    FileHandlerSegment;
 }
```

Der Zeiger »rn_Info« zeigt hier auf die Dos-Info-Struktur, die unser nächstes Ziel ist:

```
struct DosInfo
 {
  BPTR    di_McName;
  BPTR    di_DevInfo;
  BPTR    di_Devices;
  BPTR    di_Handlers;
  BPTR    NetHand;
 }
```

Hier haben wir nun den Zeiger »di_DevInfo« auf die Device-Node-Struktur, die die Information über die Devices enthält, gesetzt:

```
struct DeviceNode
 {
  BPTR    dn_Next;
  ULONG   dn_Type;
  struct MsgPort *dn_Task;
  BPTR    dn_Lock;
  BSTR    dn_Handler;
  ULONG   dn_StackSize;
  LONG    dn_Priority;
  BPTR    dn_Startup;
  BPTR    dn_SegList;
  BPTR    dn_GlobalVec;
  BPTR    dn_Name;
 }
```

Diese Struktur liefert die gesuchten Informationen über die vorhandenen Devices.

- »dn_Next« : Zeiger auf das nächste Device
- »dn_Type« : Da in der Device-Node-Struktur nicht nur Devices, sondern auch Directories und Volumes verwaltet werden, gilt als Typ
  0 für Devices,
  1 für Directories,
  2 für Disks.
- »dn_Task« : Zeiger auf den Port zum Austausch von Nachrichten
- »dn_Lock« : Zeiger auf den aktuellen Lock
- »dn_Handler« : Name des zugehörigen Handlers
- »dn_StackSize« : Größe des verwendeten Stacks
- »dn_Priority« : die zugeteilte Priorität der Devices
- »dn_Startup« : Zeiger auf eine »FileSysStartup«-Message, die noch einige Informationen über den verwendeten Handler enthält
- »dn_SegList« : Zeiger auf die Segment-Liste
- »dn_GlobalVec« : Zeiger auf den Global-Vector
- »dn_Name« : Zeiger auf den Device-Namen

Auf den obigen Informationen baut das Programm »RenDev« auf. Es erlaubt, den Namen eines vorhandenen Devices zu ändern. So läßt sich nun zum Beispiel das Printer-Device »PRT:« in »PRX:« umbenennen. Nun kann man ein Drucker-Spooler-Device als »PRT:« in das System einbinden. Danach geht die Druckerausgabe eines jeden Programms an den Drucker-Spooler und der leitet sie nacheinander an PRX: weiter, was dann dem Printer-Device entspricht.

Zuerst werden die Eingabeparameter auf Gültigkeit geprüft. Ein abschließender Doppelpunkt wird entfernt, da er vom System nicht benötigt wird. Abschließend wird noch eine Wandlung in Großbuchstaben durchgeführt. Danach hangelt man sich zur Device-Node-Struktur durch. Hier wird geprüft, ob es sich um ein Device handelt (»dn_Type = 0«). Ist dies der Fall, wird geprüft, ob es sich um das gesuchte Device handelt.

Trifft das zu, so wird sein Name geändert und das Programm beendet. Ansonsten wird das nächste Device geprüft, und zwar solange, bis die Device-Liste zu Ende ist. Wurde das gewünschte Device nicht gefunden, so wird die Device-Liste noch einmal durchgegangen und alle vorhandenen Devices zur Information des Benutzers ausgegeben. Danach wird das Programm ebenfalls beendet. Das Programm wurde in Aztec-C 3.6a geschrieben, sollte aber an andere C-Compiler leicht anzupassen sein. Der Aufruf des Programms vom CLI aus lautet

»RenDev AlterName NeuerName«,

wobei »AlterName« der Devicename, wie im System eingetragen, und »NeuerName« der gewünschte neue Name des Devices ist. Viel Spaß beim Umbenennen Ihres Devices.

(vb)

Listings

```
/*********************************************************/
/*                       RenDev                          */
/*         Programm zum Umbenennen von Devices           */
/*           (c) 1990 D.Kuntz & AMIGA DOS                */
/*              Sprache: Aztec C V.3.6a                  */
/*********************************************************/

/* Compileraufruf:  cc RenDev                            */
/* Linkeraufruf:    ln RenDev -lc                        */
```

Listing: RenDev

```


#include "exec/types.h"
#include "libraries/dos.h"
#include "libraries/dosextens.h"
#include "libraries/filehandler.h"

#define STRING(x)   (((char *)BADDR(x))+1)

extern struct DosLibrary *DOSBase;
struct DeviceNode *DeviceNode;
struct RootNode *RootNode;
```

Listing: RenDev

```
struct DosInfo *DosInfo;

main(argc, argv)
 int argc;
 char *argv[];
{
 int x;

/*     wenn ein Doppelpunkt mit eingegeben wurde,
       dann entfernen                                     */

 if ( argc == 3 )
  {
   if(argv[1][(strlen(argv[1])-1)] == ':' )
      argv[1][(strlen(argv[1])-1)] = '\0';
   if(argv[2][(strlen(argv[2])-1)] == ':' )
      argv[2][(strlen(argv[2])-1)] = '\0';
  }

    /* Programmabbruch bei falschem Aufruf */

 if ((argc != 3) || (strlen(argv[1]) != strlen(argv[2])))
  {
   printf("\nRenDev - zum Umbenennen von Devices\n");
   printf("(c) 1990 by D. Kuntz & AMIGA DOS\n");
   printf(" Aufruf : RenDev <AlterName> <NeuerName>\n");
   printf("AlterName und NeuerName muessen gleiche Laenge haben!\n\n");
   exit(FALSE);
  }

  /* Eingaben in Grossbuchstaben wandeln */

    for(x=0; x<strlen(argv[1]); x++)
      {
       argv[1][x] = toupper(argv[1][x]);
       argv[2][x] = toupper(argv[2][x]);
      }

    /* Zur DeviceInfoStruktur durchhangeln... */

 RootNode   = (struct RootNode   *) DOSBase->dl_Root;
 DosInfo    = (struct DosInfo    *) BADDR(RootNode->rn_Info);
 DeviceNode = (struct DeviceNode *) BADDR(DosInfo->di_DevInfo);

    /* ...und durchsuchen bis Device gefunden oder
       Listenende                                   */

 do
  {
   if(DeviceNode->dn_Type == (ULONG) DLT_DEVICE)
    {
    if(strcmp(STRING(DeviceNode->dn_Name),argv[1]) == 0)
     {
      strcpy(STRING(DeviceNode->dn_Name),argv[2]);
      printf("\nDevice %s: wurde in %s: umbenannt!\n\n",
                            argv[1],argv[2]);
      exit(TRUE);
     }
    }
   DeviceNode=(struct DeviceNode *)BADDR(DeviceNode->dn_Next);
  }
 while( DeviceNode!=0L );

   /* wenn das Device nicht gefunden wurde... */

   x=0;
   printf("\nKonnte Device '%s:' nicht finden! Sorry!\n\n",argv[1]);
   printf("Folgende Devices sind vorhanden:\n");
   DeviceNode = (struct DeviceNode *) BADDR(DosInfo->di_DevInfo);
    do
     {
      if(DeviceNode->dn_Type == (ULONG) DLT_DEVICE)
      /*       Finde Device        */
     {
      printf(" %s  ",STRING(DeviceNode->dn_Name));
      x++;
      if( x==5 )
       {
        printf("\n");
        x=0;
       }
     }
      DeviceNode=(struct DeviceNode *)BADDR(DeviceNode->dn_Next);
     }
    while(DeviceNode != 0L);
    printf("\n\n");
    exit(FALSE);
}

/******************* Ende von RenDev ****************/
```



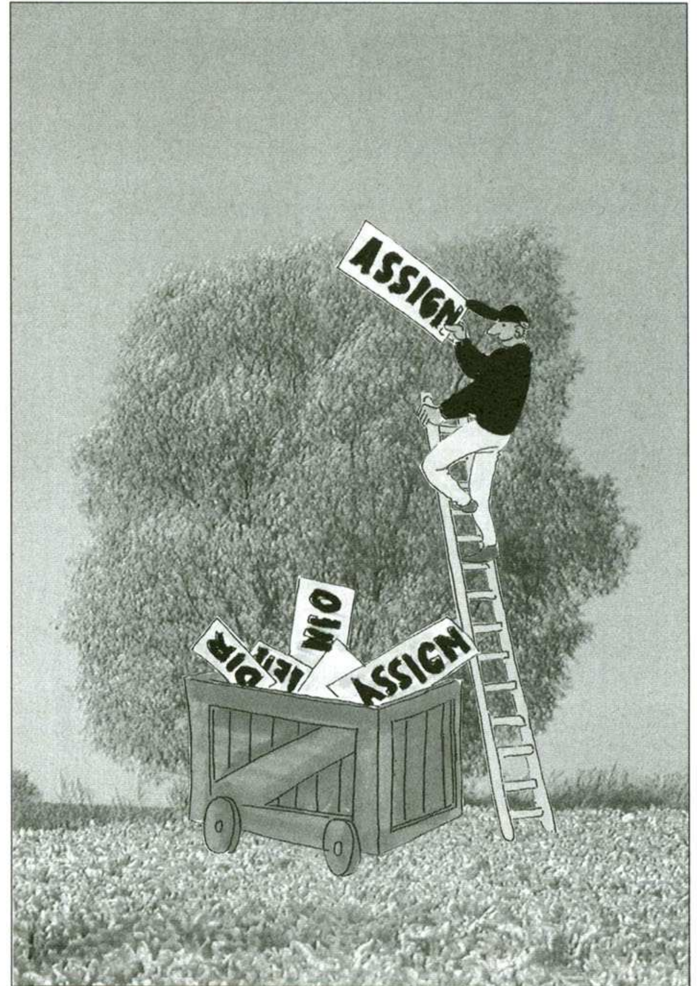
Listing: RenDev

Michael Anton

Das AmigaDOS – leicht und verständlich!

Hochstapelei mit dem CLI

In dieser Folge wollen wir uns vertieft mit den bereits in der letzten Folge kurz angesprochenen Möglichkeiten der Stapelverarbeitung befassen. Weiterhin lernen wir Wissenswertes über die verschiedenen "Geräte", die AmigaDOS kennt.



Kursfahrplan 

Teil 1 – Einführung in den CLI

Teil 2 – Was kann AmigaDOS

Teil 3 – Stapeldateien

Teil 4 – Neu dabei bei 1.3

Teil 5 – Alternativen zum CLI

Teil 6 – Multitasking

In der letzten Folge sahen wir, daß der Amiga nicht nur "echte" Programme, sondern auch eine Folge von Arbeitsanweisungen abarbeiten kann. Diese Stapeldateien sind ein wichtiges Mittel, um die Arbeit im CLI effektiv gestalten zu können, da mit ihnen durch ein einfaches Kommando eine Vielzahl von Aktionen ausgelöst werden kann. Die vorgestellte Startup-Sequence ist jedoch nur eine Form dieser Arbeit; tatsächlich lassen sich beliebige Aufgaben auf diese Art erledigen. Voraussetzung dafür ist, daß die jeweiligen Anweisungen in einer Textdatei zusammengefaßt sind.

Gestartet wird diese Datei durch den Execute-Befehl. Er bewirkt, daß die Datei (im folgenden "Script" genannt) ins Verzeichnis »T:« (für temporäre Dateien) kopiert und Schritt für Schritt abgearbeitet wird. Der Befehl »EXECUTE« steht im Normalfall in »C:«, die Scripts müssen entweder im aktuellen Verzeichnis oder in »S:« stehen, ansonsten muß der komplette Pfad angegeben werden. Ab Version 1.3 der Workbench existiert noch eine weitere Möglichkeit, solche Scripts zu starten: das Script-Flag.

Zu diesem Flag ein kleiner Exkurs in die DOS-Interna. Neben dem eigentlichen Namen einer Datei werden mit ihr auch bestimmte Attribute gespeichert, die klären, was mit der entsprechenden Datei alles gemacht werden darf. In Version 1.2 sind vier dieser Attribute vorgesehen: »R« für Read (lesen), »W« für Write (schreiben, ändern), »E« für »EXECUTE« (ausführen) und »D« für Delete (löschen). Version 1.3 führt vier zusätzliche Attribute ein, nämlich »H« für Hidden (versteckt), »S« für Script (Stapeldatei), »P« für Pure (kann mit »RESIDENT« verwendet werden – dazu in der nächsten Folge mehr) und »A« für Archive (Datei noch nicht archiviert). In der Version 1.2 führt kein Weg an der Verwendung von »EXECUTE« vorbei, unter 1.3 wird jedoch geprüft, ob bei der aufgerufenen Datei das Script-Flag gesetzt ist. Wenn dem so ist, wird automatisch »EXECUTE« gestartet, was eine erhebliche Erleichterung darstellt.

Aber wie bearbeitet man nun diese Flags? Dazu dient der Befehl »PROTECT«, der ein Setzen und Löschen dieser Informationen erlaubt. Die Syntax des Befehls ist in beiden Versionen verschieden: Generell ist der Aufruf

```
PROTECT Datei flags,
```

In der Version 1.2 bezeichnet der Parameter »flags« jene Attribute, die gesetzt werden, nicht genannte Attribute werden zurückgesetzt.

```
PROTECT Datei RW
```

würde also die Flags für Lesen und Schreiben setzen, die für Delete und Archiv zuständigen Flags jedoch zurücksetzen. Version 1.3 erlaubt darüber hinaus nicht nur das Setzen von vier weiteren Flags, sondern auch deren direkte Beeinflussung. Dazu zeigen "+" und "-" vor dem Kürzel (»RWEDHSAP«) an, ob das entsprechende Flag zu setzen ist. In der Version 1.3 setzt also

```
PROTECT Stapeljob +S
```

das Script-Flag der Datei Stapeljob, wodurch bei der Eingabe von

```
Stapeljob
```

automatisch »EXECUTE« gestartet wird – zumindest in der Version 1.3.

Was in der Theorie so gut aussieht, hat in der Praxis jedoch einen Nachteil: Sowohl Version 1.2 als auch 1.3 berücksichtigen von den RWED-Flags nur das D-Flag! Somit kann eine Datei also nur gegen Löschen geschützt werden, die drei anderen Optionen sind Makulatur. Die »HSPA-Flags« unter 1.3 sind jedoch problemlos verwendbar – sofern sie nicht von älteren Programmen gekillt werden, was leider vorkommen kann. Mit »LIST« wird übrigens ersichtlich, welche Flags bei einer Datei gesetzt sind.

Kein bedingungsloses Stapeln

Doch zurück zu unserer Einführung in die Programmierung von Scripts. Deren einfachste Form ist eine lineare Folge von Anweisungen – diese werden also nacheinander abgearbeitet.

Moderne Stapelverarbeitung erlaubt nicht nur das Abarbeiten von Teilen in Abhängigkeit von bestimmten Bedingungen, sondern auch das Umherspringen innerhalb des Scripts und eine gewisse Beeinflussung durch den Anwender und durch die Resultate anderer Befehle. Solche Strukturen kennen Sie vielleicht von Programmiersprachen wie BASIC oder C; um nichts anderes handelt es sich dabei in den Script-Dateien – sogar die Syntax weist starke Ähnlichkeiten auf.

Die bedingte Abarbeitung von Script-Teilen wird über die Befehle »IF ... ELSE ... ENDIF« ermöglicht. Die generelle Syntax lautet

```
IF Bedingung
(DOS-Befehle_1)
ELSE
(DOS-Befehle_2)
ENDIF.
```

Im Klartext heiß das nichts anderes als "Wenn die Bedingung erfüllt ist, arbeite die erste Gruppe der Befehle ab, ansonsten die andere Gruppe". Die Angabe der Alternativen ist optional, bei Bedarf genügt auch ein einfaches "Wenn – dann":

```
IF Bedingung
(DOS-Befehle)
ENDIF.
```

Allerdings können die mit diesen Befehlen erkannten Bedingungen nicht beliebig sein, stattdessen ist man auf bestimmte Fähigkeiten von DOS angewiesen. Zu diesen Fähigkeiten gehört es, daß ein Programm dem folgenden Programm Informationen über den Erfolg der eigenen Arbeit übermitteln kann. Dies wird durch sogenannte Fehlercodes bewirkt, einen numerischen Wert, der an den Nachfolger weitergegeben wird und der über Erfolg oder Mißerfolg des Programms Aufschluß gibt. Dieser Wert wird als eine Zahl übergeben, wobei "0" nach Konvention eine problemlose Abarbeitung signalisiert. Je nach Programm können jedoch auch andere Werte geliefert werden, die über die Schwere des Fehlers Auskunft geben. Diese Schwere wird in drei Gruppen gegliedert, der auch die Schlüsselwörter für die Bedingung entsprechen. Bei einem Fehlercode zwischen 1 und 5 handelt es sich um eine Warnung (»WARN«) – es traten kleinere Komplikationen auf. Ein Fehlercode von 10 oder mehr (»ERROR«) zeigt an, daß der letzte Befehl erhebliche Probleme hatte, Werte über 20 (»FAIL«) besagen, daß die letzte Aktion gehörig daneben ging. Während die ersten Fehler noch abgefangen werden können, führt ein »FAIL« auch zum Abbruch des jeweiligen Scripts. Die "Toleranzgrenze" kann jedoch mit »FAILAT x« eingestellt werden, da einige Programme auch bei "harmlosen" Fehlern hohe Fehlercodes liefern. Diese Änderung gilt jedoch nur innerhalb eines Scripts und sollte dort so früh wie möglich definiert werden. Die anderen Bedingungen bestehen in der Existenz einer bestimmten Datei (»EXISTS Datei«) oder der Gleichheit zweier Zeichenketten (»Text1 EQ Text2«). Ab Version 1.3 können auch numerische Werte miteinander verglichen werden, wobei neben der Gleichheit (»EQ«) auch die Bedingungen "Größer als" (»GT«) oder "Größer gleich" (»GE«) erfragt werden können. Die Option »NOT« invertiert in allen Fällen die Bedingung, so daß Abfragen wie »IF NOT EXISTS Datei« (= Datei nicht vorhanden) oder »IF NOT Wert1 EQ Wert2« (Wert1 ungleich Wert2) möglich sind.

Beim Stapeln springen

Neben diesen bedingten Arbeiten kann man in einem Script auch umherspringen, man muß also nicht alle Zeilen nacheinander abarbeiten. Zu diesem Zweck können Sprungmarken, auch Labels genannt, definiert werden. Bei einem Sprung auf ein solches Label wird ganz einfach die darauffolgende Zeile im Script ausgeführt. Zur Definition dient der Befehl »LAB«, hinter dem eine Zeichenkette angegeben werden muß. Dieses Label läßt sich im Script über »SKIP« anspringen:

```
(wird_gemacht)
SKIP Weiter
(wird_nie_gemacht)
LAB Weiter
(weiter_gehts).
```

Bei diesem Sprung wird der Befehl »wird_nie_gemacht« einfach übersprungen. Das Beispiel ist natürlich nicht ideal, normalerweise wird dieser Sprung in Verbindung mit Bedingungen eingesetzt. Aber Vorsicht: Version 1.2 erlaubt nur die "Flucht nach vorn", Rücksprünge sind erst ab Version 1.3 möglich, diese werden durch die Option »BACK« markiert:

```
LAB PlayItAgain
PlayIt SKIP PlayItAgain BACK
```

Dieses Beispiel zeigt auch gleich die Gefahr, die bei solchen Sprüngen auftritt, wir haben soeben eine klassische Endlosschleife produziert... In solchen Konstrukten sollte also immer eine Abbruchbedingung vorgesehen werden, sonst hilft nur [Ctrl + D]! Diese Bedingung muß zu einem Befehl hinter dem »SKIP BACK« führen, wenn andere Befehle noch folgen sollen; für den Abbruch gibt es auch noch eine andere Möglichkeit.

Soll ein Script vorzeitig beendet werden, so kann dazu der Befehl »QUIT« verwendet werden. Diesem Befehl kann zusätzlich ein Fehlercode übergeben werden (zum Beispiel »QUIT 255«), der anderen Scripts oder dem Anwender weitere Informationen liefern kann. Im folgenden Beispiel wird ein Script abgebrochen, wenn eine bestimmte Datei nicht vorhanden ist:

```
IF NOT EXISTS Prog_Datei
ECHO "Datenfile fehlt..."
QUIT
ENDIF
Do_Programm
```

Die Stimme des Herrn

Idealerweise sollte ein Script auch auf verschiedene Wünsche des Anwenders Rücksicht nehmen, und das macht es auch...

So kann ein Script fragen, ob der Anwender dieses oder jenes bevorzugt, dazu dient der Befehl »ASK«. Er stellt eine beliebige Frage, die entweder mit "Y" für "Ja" oder "N" für "Nein" beantwortet werden darf. Bei der Antwort zählt nur der erste Buchstabe, ist dieser "N" oder ein einfaches "Enter«, so liefert »ASK« einen Fehlercode von 0, ein "Y" oder "Ya" oder "Yes" liefert einen Fail-Wert, der via IF abgefragt werden kann. Ein Beispiel:

```
ASK "Weitermachen (Y/N)"
IF NOT FAIL
QUIT
ENDIF
Mache_weiter
```

Oder zurück zu »SKIP«. Hier noch eine Variante für die Version 1.3:

```
LAB PlayItAgain
PlayIt
ASK "Nochmal (Y/N)"
IF NOT FAIL
QUIT
ENDIF
SKIP PlayItAgain BACK
```

Das ist jedoch noch nicht alles. Einem Script können beim Aufruf auch noch zusätzliche Parameter übergeben werden, die dann intern weiterverwendet werden. Das Prinzip ist beispielsweise vom AmigaBASIC bekannt: »AMIGABASIC Name« lädt nicht nur das BASIC, sondern auch das Programm »Name« und führt es aus. Ähnlich verhält es sich mit Scripts: Auch ihnen können mehrere Parameter übergeben werden, allerdings sind noch einige Interna zu beachten. Den Parametern müssen im Script Variablennamen zugeordnet werden, was durch die Anweisung ».KEY« oder ».K« geschieht. Soll im Script auf diese Variable zugegriffen werden, so muß der Name in Klammern angegeben werden. Die Art der Klammerung kann mit den Kommandos ».BRA« und ».KET« verändert werden. Normalerweise ist die Form <variable> vorgegeben, mit ».BRA [« und ».KET]« kann innerhalb des Scripts jedoch auch die Form [variable] oder ähnliches gewählt werden. C-Freaks werden vielleicht {...} bevorzugen...

Den Variablen kann auch noch ein Eingabezwang zugewiesen werden: Steht in der Key-Anweisung hinter dem Namen »/A«, so muß dieser Wert unbedingt übergeben weden, bei »/S« ist die Angabe optional. Allerdings können Variablen auch noch innerhalb des Scripts zugewiesen werden. Wird beim Aufruf eines Scripts für einen erwarteten Parameter keine Angabe gemacht, so läßt sich für diese Position über »$« eine Vorgabe einrichten, mit ».DEF« können auch beliebige Variablen definiert werden. Analog zu ».BRA« und ».KET« kann das Symbol für die Zwangszuweisung auch mit ».DOL« geändert werden, also beispielsweise ».DOL =«, um bei Vorgaben »$« durch » = « zu ersetzen.

Noch ein Wort zur Gestaltung der Scripts. Solange sie reine ASCII-Texte sind, ist es egal, wie sie aussehen – im Interesse der Lesbarkeit sollte man jedoch einige Konventionen beachten. Am besten hält man sich an Konventionen "normaler" Programmiersprachen

wie Pascal und C, was die Strukturierung des Textes betrifft. Die Einrückung der Befehle in If-Endif-Blöcken sieht nicht nur "gut" aus, sie erleichtert auch das Suchen von Fehlern. Kommentare sind ebenfalls eine sinnvolle Angelegenheit, sie werden durch ";" eingeleitet. Das Semikolon dient als Trennzeichen, das sowohl allein in einer Zeile als auch nach einem Befehl stehen kann:

```
; Dies ist ein Kommentar
DIR ; Dieses auch...
```

Alles was nach ";" kommt, wird bis zum Ende der Zeile ignoriert, auch wenn es sich um normale Aufrufe handelt. Das ist bei Tests recht praktisch, da probehalber die Ausführung unterbunden werden kann. Das letzte Script zeigte noch das Directory an, aber bei

```
;DIR ; Zeige Inhalt
```

tut sich erst dann was, wenn das erste Semikolon entfernt wird. An dieser Stelle möchten wir das Thema Stapelverarbeitung vorerst ruhen lassen. Ihre Kreativität ist nun gefragt – machen Sie aus dem bisher Gelernten das Beste und schauen Sie, wie Sie Scripts in der täglichen Arbeit einsetzten können. Abbildung 2 zeigt ein größeres Script, welches den Inhalt dieser Folge nochmals in der Praxis zeigt.

Bis jetzt haben wir uns mit Laufwerken, Dateien und Eingabezeilen beschäftigt – deren Verwaltung ist sicher die Hauptaufgabe von DOS. Es gibt jedoch auch noch andere Dinge, um die sich ein Betriebssystem kümmern kann oder soll. Ein jeder Computer hat, sofern er nicht ausschließlich zum Spielen verwendet wird und der Etat es zuläßt, einen Drucker. Den kann man nicht nur innerhalb von Anwendungen benutzen, sondern durchaus auch direkt unter DOS – beispielsweise um seine Scripts zu Papier zu bringen. Zum Ansprechen solcher Peripheriegeräte stellt DOS bestimmte Gerätenamen zur Verfügung, diese Geräte sind zumeist mit einer bestimmten Schnittstelle gekoppelt. Zwei solcher Anschlußmöglichkeiten sind beim Amiga gegeben: die parallele (Centronics-) und die serielle (RS 232-) Schnittstelle. Diesen physikalischen Geräten weist das DOS Namen zu, unter denen sie dem Betriebssystem bekannt sind. Entsprechend der Konvention haben sie mindestens drei Buchstaben und einen Doppelpunkt, wie wir das schon von physikalischen oder logischen Laufwerken her kennen. So heißt die parallele Schnittstelle »PAR:« und die serielle analog »SER:« – das leuchtet ein, oder?

Die "Gerätchen-Frage"

Aber diese Geräte sind noch nicht alles, was das DOS kennt. So gibt es noch ein Gerät namens »PRT:« – das ist nichts anderes als der Drucker, der mit »PREFERENCES« als Systemdrucker installiert wurde. Das schöne an »PRT:« ist, daß dieser Drucker auf jeden Fall über »PRT:« angesprochen wird, egal an welcher Schnittstelle er nun hängt. Maßgebend hierfür ist die Auswahl von »PREFERENCES«; »PRT:« ist also kein "physikalisches", sondern ein "logisches" Gerät. Er gleicht hierin den mit »ASSIGN« erzeugten "Laufwerken": Es gibt sie eigentlich nicht, aber sie sind dennoch "bekannt". Im Kontrast zu »SER:« und »PAR:« ist »PRT:« auch mit einer Eigenintelligenz ausgestattet, die sich durch die Umsetzung von Steuersequenzen bemerkbar macht. Diese hatten wir in der letzten Folge für die Bildschirmsteuerung bereits kennengelernt. Während eine solche Sequenz, sagen wir mal »*e[3m« für Kursivdruck, bei der Ausgabe über »PAR:« oder »SER:« nur Klartext, also »*e[3m« erzeugt, wird bei der Ausgabe über »PRT:« tatsächlich der Drucker auf Kursivschrift umgeschaltet.

Tabelle 1 zeigt eine Übersicht über die Steuerzeichen, mit denen ein Drucker am Gerät »PRT:« eingestellt werden kann. Die Steuersequenzen werden vom jeweiligen »PREFERENCES« ausgewählten Druckertreiber in die entsprechenden Steuerzeichen für den gewählten Drucker übersetzt.

Praktisch: die Escape-Sequenzen

Dabei ist allerdings zu beachten, daß nicht jeder Drucker über alle angeführten Möglichkeiten verfügt. Die Befehle für die Farbwahl würden bei einem Schwarzweißdrucker entweder schon im Treiber ausgefiltert oder spätestens im Drucker selbst ignoriert oder falsch interpretiert.

Eingeleitet werden die Sequenzen durch das Zeichen »ESC«. Dieses erhält man durch Drücken der Taste [Esc], worauf eine invertierte eckige Klammer auf dem Bildschirm angezeigt wird.

Falls ein Editor diese Art der Eingabe von Sonderzeichen nicht unterstützt, kann alternativ auch die Folge »*e« oder »*E« verwendet werden. Während es in diesem Fall egal ist, ob der Buchstabe groß oder klein geschrieben wird, müssen die übrigen Zeichen in der Sequenz genau übernommen werden.

Der Buchstabe "n" in einer Sequenz steht für einen numerischen Wert, dessen Bedeutung kommentiert wird. Um die Seitenlänge über »{ESC}[nt« auf 70 Zeilen pro Seite einzustellen, würde die Sequenz entweder »[[70t« oder »*e[70t« lauten.

Sequenz	Aufgabe
{ESC}c	Reset
{ESC}#1	Initialisierung
{ESC}D	Zeilenvorschub
{ESC}E	Wagenrücklauf & Zeilenvorschub
{ESC}M	Zeilenrückschub
{ESC}[0m	Normaler Zeichensatz
{ESC}[3m	Kursiv ein
{ESC}[23m	Kursiv aus
{ESC}[4m	Unterstrichen ein
{ESC}[24m	Unterstrichen aus
{ESC}[1m	Fettschrift ein
{ESC}[2m	Fettschrift aus
{ESC}[nm	Vordergrundfarbe (n=31..38)
{ESC}[nm	Hintergrundfarbe (n=41..48)
{ESC}[0w	Normale Zeichenbreite (Pica, 10 cpi)
{ESC}[2w	Elite ein (12 cpi)
{ESC}[1w	Elite aus
{ESC}[4w	Condensed ein (Kompaktschrift, 17 cpi)
{ESC}[3w	Condensed aus
{ESC}[6w	Enlarged ein (Breitschrift)
{ESC}[5w	Enlarged aus
{ESC}[6"z	Schattenschrift ein
{ESC}[5"z	Schattenschrift aus
{ESC}[4"z	Doublestrike ein (Doppelanschlag)
{ESC}[3"z	Doublestrike aus
{ESC}[2"z	NLQ ein (Schönschrift)
{ESC}[1"z	NLQ aus
{ESC}[2v	Hochstellen ein
{ESC}[1v	Hochstellen aus
{ESC}[4v	Tiefstellen ein
{ESC}[3v	Tiefstellen aus
{ESC}[0v	Zeile normal darstellen
{ESC}L	Zeile teilweise hochstellen
{ESC}K	Zeile teilweise tiefstellen
{ESC}(B	Amerikanischer Zeichensatz
{ESC}(R	Französischer Zeichensatz
{ESC}(K	Deutscher Zeichensatz
{ESC}(A	Britischer Zeichensatz
{ESC}(E	Dänischer Zeichensatz I
{ESC}(H	Schwedischer Zeichensatz
{ESC}(Y	Italienischer Zeichensatz
{ESC}(Z	Spanischer Zeichensatz
{ESC}(J	Japanischer Zeichensatz
{ESC}(6	Norwegischer Zeichensatz
{ESC}(C	Dänischer Zeichensatz II
{ESC}[2p	Proportionalschrift ein
{ESC}[1p	Proportionalschrift aus
{ESC}[0p	Proportionalschrift löschen
{ESC}[n E	Proportional-Offset (n) setzen
{ESC}[5 F	Linksbündig justieren
{ESC}[7 F	Rechtsbündig justieren
{ESC}[6 F	Blocksatz
{ESC}[1 F	Zentrieren
{ESC}[0 F	Keine Justierung
{ESC}[3 F	Nach Zeichenbreite justieren
{ESC}[0z	8 Zeilen / Zoll
{ESC}[1z	6 Zeilen / Zoll
{ESC}[nt	n Zeilen / Seite
{ESC}[nq	n Zeilen bei Perforation überspringen
{ESC}[0q	Kein Perforations-Sprung
{ESC}#9	Linken Rand anfahren
{ESC}#0	Rechten Rand anfahren
{ESC}#8	Oberen Rand anfahren
{ESC}#2	Unteren Rand anfahren
{ESC}[n1;n2r	Oberen und unteren Rand einstellen
{ESC}[n1;n2s	Linken und rechten Rand einstellen
{ESC}#3	Ränder löschen
{ESC}H	Horizontaler Tabulator
{ESC}J	Vertikaler Tabulator
{ESC}[0g	Horizontalen Tabulator löschen
{ESC}[3g	Alle horizontalen Tabulatoren löschen
{ESC}[1g	Vertikalen Tabulator löschen
{ESC}[4g	Alle vertikalen Tabulatoren löschen
{ESC}#4	Alle Tabulatoren löschen
{ESC}#5	Standard-Tabulatoren setzen
{ESC}[Pn"x	Erweiterten Befehl (n) senden

Abbildung 1. Eine 'lineare' Stapeldatei...

Das sind jedoch nicht alle Schnittstellen, die DOS kennt. Mit einem "ganz normalen" Gerät arbeiten wir beispielsweise seit Beginn des Kurses – es ist so normal, daß man es fast schon wieder übersieht. Die Rede ist von Tastatur und Bildschirm, im EDV-Jargon werden diese Bedienelemente auch als "Konsole" bezeichnet. AmigaDOS vergibt für die Kombination aus beidem den Namen »CON:«, aber auch der Name »RAW:« ist dafür nicht unbekannt. Der Unterschied zwischen »CON:« und »RAW:« ist ähnlich wie der zwischen »PRT:« und »PAR:« – in einem Fall werden bestimmte Daten nochmals gesondert interpretiert, im anderen ganz einfach unbearbeitet weitergeleitet. Ab Version 1.3 gibt es eine neue Art "Konsole", nämlich »NEWCON:«. Dieses "Gerät" unterscheidet sich vor allem in der Bearbeitung der Kommandozeile von »CON:« – in der neuen Version kann beispielsweise die Eingabe nachträglich bearbeitet werden.

»NEWCON:« ist jedoch nicht das einzige neue Gerät, mit dem Version 1.3 aufwarten kann. Da wäre beispielsweise »AUX:« – im Prinzip nichts anderes als »SER:«, allerdings werden hier die Daten direkt an das angeschlossene Gerät weitergegeben und nicht zwischengespeichert. Weniger praktikabel, aber durchaus interessant ist das Gerät »SPEAK:« – der Lautsprecher. Alle Informationen, die an dieses Gerät geschickt werden, erschallen im Klartext über die Sprachausgabe, ähnlich wie beim Programm »SAY«. Ein anderes, erst ab 1.3 verfügbares Gerät ist »PIPE:«, das gleichzeitig ablaufende Programme miteinander kommunizieren läßt. Zu diesem Punkt jedoch an anderer Stelle mehr.

Etwas aus der Reihe fallen die in beiden Versionen vorhandenen Geräte »*« und »NIL:«. Während sich allen bisher besprochenen Geräten immerhin noch eine Buchse im Gehäuse des Amiga zuordnen läßt, wird es bei diesen beiden Geräten schon etwas problematischer. Und doch ist alles ganz einfach. »*« ist nichts anderes als das gerade aktive CLI-Fenster. »NIL:« ist ganz einfach Niemandsland. Informationen, die an »NIL:« gesendet werden, verschwinden auf Nimmerwiedersehen im Amiga, von dieser Stelle darf jedoch auch keine sinnvolle Information erwartet werden. Das Gerät »NIL:« ist ganz einfach ein Entsorgungsgerät – was dorthin geht, war von jeher zu vernachlässigen.

Achtung, Umleitung...

Aber was fangen wir nun mit diesen vielen Geräten an? Wir haben es bereits angedeutet, man kann solchen Geräten entweder Informationen zukommen lassen oder von dort Informationen entnehmen. Solche Geräte verhalten sich im Prinzip nicht anders als ein ganz normales Diskettenlaufwerk. Man kann Daten dorthin kopieren oder von dort entgegennehmen. Für AmigaDOS ist der Unterschied auch gar nicht so groß. Das liegt wohl auch daran, daß der Datenfluß zwischen zwei Geräten tatsächlich nicht mehr als ein Strom von Bits und Bytes ist, der sehr leicht in andere Kanäle – und damit auf andere Geräte – umgeleitet werden kann. Die EDV spricht hier von Pipes oder Pipelines, also Röhren oder Rohrleitungen, die den Datenfluß ganz einfach kanalisieren und umleiten. Das muß dem DOS natürlich mitgeteilt werden. Auf dem Amiga haben sich dafür zwei Wege etabliert, die gleichberechtigt sind; einer der Wege lehnt sich auch stark an Methoden "normaler" Betriebssysteme wie MS-DOS oder UNIX an, der andere verfolgt die etwas ausführlichere Angabe von Optionen nach AmigaDOS.

Greift man auf das Bild des Flusses der Daten von einem Programm zum anderen oder von einer Datei zu einer anderen zurück, so sind zwei Richtungen offensichtlich: "Von" und "Nach". Auf englisch heißen diese Richtungen »FROM« und »TO«, zwei Schlüsselwörter, die AmigaDOS bei vielen Befehlen unterstützt. Alternativ kann der Datenfluß auch durch die Symbole "<" und ">" sichtbar gemacht werden – die Richtung des Pfeiles gibt auch die Richtung des Datenflusses an. Die Verwendung der "Keywords" ist weitgehend freigestellt, wer bereits unter MS-DOS oder UNIX gearbeitet hat, wird jedoch die "symbolische" Schreibweise vorziehen.

Diese Umleitung kann im CLI mehrfach verwendet werden. Zum ersten ist diese Form natürlich eine einfache Art, die Ausgabe von Programmen umzuleiten. Diese Umleitung geht nicht nur auf andere Geräte, sondern auch auf Dateien. Wer das Directory einer Diskette für die Verwendung innerhalb einer Textverarbeitung benötigt, kann mit

```
DIR > Inhalt.Txt
```

ein entsprechendes Text-File einfach erstellen und dieses dann als Block einlesen. Wer statt dessen einen Ausdruck benötigt, der ist mit

```
DIR > PRT:
```

gut beraten. Oder möchten Sie den Inhalt einer Diskette lieber hören statt sehen?

```
DIR > SPEAK:
```

löst das Problem (ab Version 1.3).

Die Optionen »TO / FROM« und »> / <« sind überwiegend synonym, weiterhin ist zu beachten, daß AmigaDOS beim Fehlen von Schlüsselwörtern auf Schablonen zurückgreift. So bezeichnet der erste Name beim Fehlen der Keywords »FROM« und »TO« die Quelle, der zweite das Ziel. Wer mit anderen Betriebssystemen vertraut ist, der schätzt das, da das dort der Normalfall ist.

```
COPY Datei PRT:
```

ist äquivalent zu

```
COPY TO PRT: <Datei
```

– die Wege des AmigaDOS sind für manchen zwar etwas sehr merkwürdig, das Resultat ist jedoch akzeptabel.

Zu Informations- und Dokumentationszwecken kann es durchaus sinnvoll sein, die Ausgabe eines Programms in eine Datei umzuleiten – beispielsweise, wenn diese Informationen noch anderweitig benötigt werden. "Überflüssige" Ausgaben eines Programms können jedoch genausogut ins Daten-Nirwana (besser "NIL-Wana"?) geschickt werden, wenn die Ausgabe der Informationen eines Programms uninteressant ist. Teilweise ist das sogar notwendig, wenn ein Programm meint, es müsse ein Fenster für die Ausgabe von Texten offenhalten. Dieses Fenster wäre meist das aktuelle CLI-Fenster, es wäre dann aber für weitere Aktionen blockiert. Lenkt man die zu erwartenden Ausgaben um, beispielsweise mit

```
TuWas >NIL:,
```

verlaufen dann alle Ausgaben von »TuWas« "im Sande" und das CLI bleibt für weitere Aktionen frei. Könnten die Ausgaben dennoch interessant sein, kann man sie notfalls auch auf andere Geräte wie »PRT:« oder gar »SPEAK:« umleiten – letzteres eine ganz andere Art der Kommunikation... Zwingend notwendig kann diese Umleitung auch bei der Arbeit mit mehreren Programmen oder Hintergrundprogrammen sein, aber davon mehr in der Folge über Multitasking. Beispiele finden sich in den "offiziellen" Startup-Sequences.

Die direkte Übergabe von Eingaben an eine Datei kann auch hilfreich sein, um Scripts zu erstellen. Für die Erstellung eines Scripts ist es nicht immer nötig, einen separaten Texteditor zu starten,

```
COPY * Script
```

leistet ähnliche Dienste. Nach diesem Kommando können Script-Sequenzen über die Kommandozeile eingegeben werden. Sind alle Eingaben komplett, kann die Eingabe mit [Ctrl + \] beendet werden, eine Datei mit dem Inhalt der Eingabe wird automatisch erstellt und kann mit »EXECUTE« Script gestartet werden. Die Auswertung von [Ctrl + \] ist einer der Punkte, der »CON:« oder »NEWCON:« von »RAW:« unterscheidet. Dafür bietet »RAW:« jedoch auch andere Vorteile: Die Rohdaten gelangen direkt und unverfälscht an den Computer während bei »CON:« oder »NEWCON:« erst noch jede Eingabe mit [Enter] bestätigt werden muß, können über »RAW:« auch einzelne Tasten sofort ausgewertet werden.

Mit dem Wissen über die Wege der Umleitung wird übrigens auch der Sinn der Befehle ».BRA« und ».KET« in Scripts deutlich. Normalerweise werden durch die Vorgaben für den Variableninhalt auch die Zeichen für die Umleitung blockiert. Durch eine entsprechende Umbenennung können somit die jeweiligen Symbole für eine Umleitung freigehalten werden. Es sei jedoch empfohlen, für eine gute Übersichtlichkeit die Befehle »TO« und »FROM« beizubehalten.

Für diese Folge soll es genug der Tipperei sein. Wenn Sie schon unter Maus-Entzug leiden, so laden Sie möglichst schnell die Workbench und trainieren Sie statt der Finger die Armmuskeln...

(cd)

Aufgabe

Auch diesmal gibt es wieder keine besonderen Hausaufgaben. Vielleicht haben Sie diese ja auch seit der ersten Folge in schlechter Erinnerung... "Spielen" Sie statt dessen mit den neuen Befehlen für die Programmierung von Scripts. Basteln Sie sich neue und für Ihre Anwendungen praktikable Scripts, oder erweitern Sie Ihre persönliche Startup-Sequence. Ein Hinweis: Je nach Anwendung, die gestartet werden soll, können Vorarbeiten nötig sein – die Definition von »ASSIGNs« oder das Laden von Tools. Im einfachsten Fall kann der Rechner auch nur fragen, ob Sie heute mit der Workbench oder mit dem CLI arbeiten wollen.

Experimentieren Sie auch mit den verschiedenen Geräten des AmigaDOS und den jeweiligen Umleitungsmöglichkeiten und ihren Konsequenzen. Lassen Sie Ausgaben von Programmen oder den Inhalt von Dateien auf Papier, in Dateien oder in Ihrem Gehörgang landen. Und vergessen Sie bei Ihrer Arbeit eines nicht: Computer sind dümmer als Menschen – aber schneller...

Listings

```
Beispiel  1:  Ein einfaches lineares Script; Es definiert
einige  logische  Laufwerke;  auf einer Festplatte und zeigt
sie    an...

ECHO        "*N*NDefiniere        Laufwerke        für        die
Programmiersprachen...*N"
ASSIGN ABasic   : DH0:Sprachen/Basic/AmigaBasic
ASSIGN GFABasic: DH0:Sprachen/Basic/GFABasic
ASSIGN KPascal : DH0:Sprachen/Pascal/KickPascal
ASSIGN PCQ      : DH0:Sprachen/Pascal/PCQPascal
ASSIGN ASM      : DH0:Sprachen/Assembler/Devpack

ASSIGN  LIST

ECHO "*N*E[33mViel Spaß beim Programmieren...*E[0m"
```

```
; Beispiel 2: Script mit Eingriffsmöglichkeiten für den Anwender.
; Es wird das aktuelle Directory auf verschiedenen Geräten ausgegeben.
; Es demonstriert auch den Einsatz von Steuersequenzen.


ECHO "*ec*e[4mWir geben das aktuelle Directory aus...*e[0m*N*N"

ASK "Ausgabe auf den Bildschirm?    -- (Y)a / (N)ein"
IF WARN
  ECHO "*N*e[2m"
  DIR
  ECHO "*e[0m"
  SKIP Ende
ENDIF

ASK "*NAusgabe auf den Drucker?        -- (Y)a / (N)ein"
IF WARN
  ASK "*N*e[33mBitte Drucker bereitmachen und ENTER drücken"
  ECHO "*e[32mIch drucke...*e[0m"
  ECHO >PRT: "*ec*N*e[4mInhalt der Diskette:*e[24m*N*N"
  DIR >PRT:
  ECHO >PRT: "*N*N"
  SKIP Ende
ENDIF

ASK "*NAusgabe in eine Datei?          -- (Y)a / (N)ein"
IF WARN
  ECHO "*N*e[33mIch schreibe in Datei Inhalt.Txt...*e[32m*N"
  DIR >Inhalt.Txt
  TYPE Inhalt.Txt
  ECHO "*e[0m"
  SKIP Ende
ENDIF

ASK "*NSoll ich vorlesen?              -- (Y)a / (N)ein"
IF WARN
  ECHO >SPEAK: "Here we go!"
  DIR >SPEAK:
  ECHO >SPEAK: "Thats all!"
  SKIP Ende

  ; Eine Alternative für Version 1.2, sofern SAY verfügbar ist
  ; Entsprechende Varianten notfalls  "auskommentieren"...
  ; (Über SPEAK: klingt es wesentlich besser!)

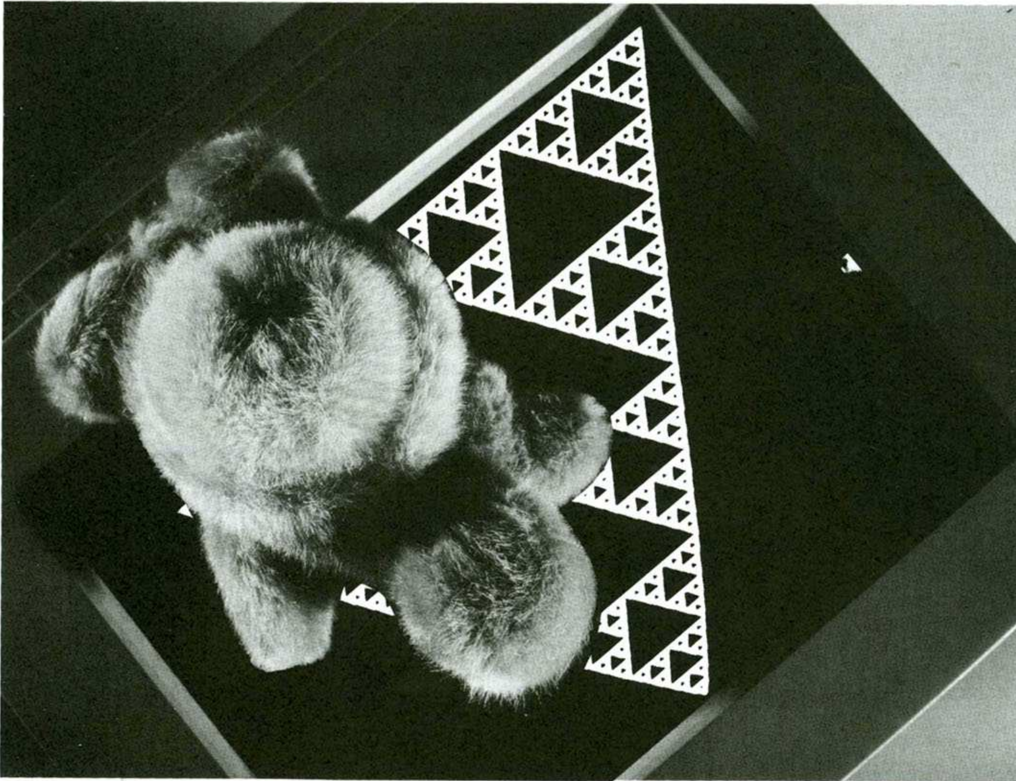
  ;DIR >T:Dirliste.Tmp
  ;SAY ¦ Here we go!
  ;SAY -x T:Dirliste.Tmp
  ;SAY ¦ Thats all!
  ;SKIP Ende
ENDIF

ECHO "*N*NJa was denn sonst noch...*N"
QUIT

LAB Ende
ECHO "*N*NEs war mir ein Vergnügen...*N"
```



Listing Startup-Sequenz



M. Cordes und N. Nebel

Turtle-Grafik

Von der Arbeit einer Schildkröte

Vom kindgemäßen Arbeiten bis hin zum komplexen Programmieren reicht die Spannbreite der Turtle-Grafik. Wie sich die Schildkröte mit der Rekursion verträgt, zeigen unsere Beispiele.

Bekannt geworden ist die Turtle-Grafik durch die Programmiersprache LOGO, die vor einigen Jahren speziell als Computersprache für Kinder entwickelt wurde. Das Ziel dieser Programmiersprache war es, Kinder mit möglichst einfachen Befehlen an den Computer, genauer gesagt an das Programmieren, heranzuführen. Die Turtle-Grafik zielt dementsprechend darauf ab, eine einfache Handhabung von Grafikfunktionen zu ermöglichen. Zu diesem Zweck definiert man einen Grafik-Cursor in Form einer Schildkröte, die sich über die Zeichenfläche bewegt und dabei Linien hinterläßt. Durch rekursive Programmierung kann man dann mit wenigen Befehlen eindrucksvolle Bilder erzeugen, die auch fortgeschrittene Programmierer faszinieren können. Die Beispielprogramme "Dreieck", "Busch" und "Farn" machen das deutlich. Wie ist solch ein Programm aufgebaut?

Greifen wir das Beispiel "Farn" heraus: Der Farn besteht lediglich aus drei Rechtecken. Jedes einzelne Rechteck besteht wiederum aus drei Rechtecken. Diese wiederum... Gibt man bei der Ausführung des Programms die Rekursionstiefe entsprechend niedrig an, läßt sich die ursprüngliche geometrische Figur erkennen. Je größer die Schachtelungstiefe wird, desto feiner und filigraner fällt das Ergebnis aus. Optimale Ergebnisse wurden in unseren Bildern je nach Komplexität der Grundfigur mit Schachteltiefen zwischen 9 und 25 erreicht (vergleiche Bild 1 und Bild 2).

Wie funktioniert eigentlich die Turtle? Der Benutzer hat die Möglichkeit, ihre Bewegung durch recht einfache Befehle zu kontrollieren. Will man beispielsweise einen Kreis auf dem Bildschirm erzeugen, so läßt man die Turtle viele kleine Schritte vorwärts machen. Dabei wird nach jedem Schritt eine kleine Drehung vollzogen. Die Befehle sind als Funktionen für C-Compiler geschrieben, so daß Grafik innerhalb eines C-Programms einfach erzeugt werden kann. Schauen wir uns die Befehle im einzelnen an.

● »SETPOS«
Argumente: X und Y, beides "double"-Werte (Fließkommazahlen)
Erläuterung: Setzt die Turtle auf die angegebene Position auf dem Bildschirm (angegeben in normalen Bildschirmpunkten, allerdings intern als Fließkommazahlen gespeichert, um die Genauigkeit beim Ausführen mehrerer Befehle zu erhöhen).
Erlaubter Wertebereich:
X: 0 bis 639
Y: 0 bis 255

● »SETHEADING«
Argument: Winkel als Integerzahl
Erläuterung: Setzt den Winkel der Turtle auf den angegebenen Wert (in Grad). Der Winkel 0 zeigt dabei nach oben, 90 nach rechts, etc.
Erlaubter Wertebereich:
Winkel: von 0 bis 359 Grad

● »PENUP«
Argumente: keine
Erläuterung: Bewirkt ein Anheben des Zeichenstiftes, bei nachfolgenden Bewegungen wird nicht gezeichnet. Das dient dazu, die Zeichnung an einer anderen Stelle fortzusetzen.

● »PENDOWN«
Argumente: keine
Erläuterung: Gegenteil von »PENUP«. Bewirkt ein Absenken des Zeichenstiftes, so daß

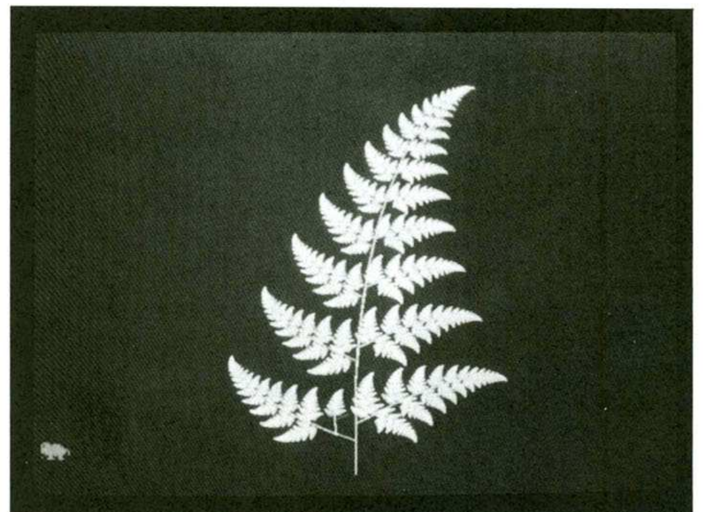


Bild 1. Mit der Turtle-Grafik lassen sich recht interessante Bilder erzeugen. Wie hier zu sehen der Farn ...

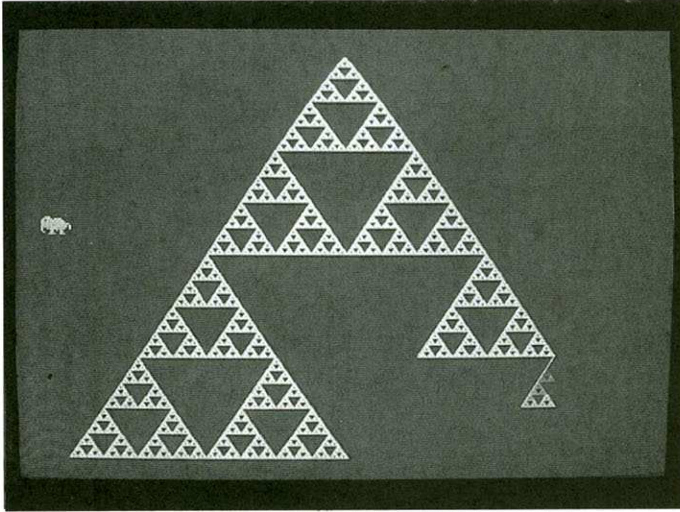


Bild 2. ... oder der Sierpinsky-Tetraeder

bei nachfolgenden Bewegungen wieder gezeichnet wird. Dieser Befehl muß auch am Anfang eines Programms eingesetzt werden, wenn gezeichnet werden soll.

● »FD«

Argument: Länge als "double"-Wert

Erläuterung: FD steht für forward, also vorwärts. Dieser Befehl bewegt die Turtle um den angegebenen Weg vorwärts, je nach »PENUP/PENDOWN« wird dabei eine Linie gezeichnet.

● »BACK«

Argument: Länge als "double"-Wert

Erläuterung: Dieser Befehl arbeitet genauso wie »FD«, nur daß die Turtle sich nicht vorwärts, sondern rückwärts bewegt.

● »PUSHT«

Argumente: keine

Erläuterung: Speichert den momentanen Zustand der Turtle (Position und Winkel) auf einem internen Stack, von wo er später mit »POPT« wieder zurückgeholt werden kann.

● »POPT«

Argumente: keine

Erläuterung: Holt einen abgelegten Zustand vom Stack zurück. »PUSHT« und »POPT« dienen dazu, die Ausführung einer Zeichnung zu unterbrechen, etwas anderes zu zeichnen, um dann an der Stelle fortzufahren, wo unterbrochen wurde. Das Programm »Busch« dient hier als gutes Beispiel.

Wer über einen 24-Nadel-Drucker mit 360 dpi sowie 2 MByte RAM-Speicher verfügt, kann die Turtle-Befehle in ähnlicher Art auch auf DIN-A4-Seiten in höchster Auflösung (2880x3960 Punkte) anwenden. Hier ist aber der Programmierer gefragt, denn es sind einige Änderungen an den Assembler-Routinen notwendig, besonders muß eine neue Linienfunktion definiert werden. Probieren Sie doch ruhig ein wenig aus, und finden Sie Ihren eigenen Algorithmus.

(vb)

AMIGA DOS Info

Wer sich die Änderungen der Assembler-Routinen nicht zutraut, kann eine fertig angepaßte Version für den NEC P6 zusammen mit einem Programm zum Ausdrucken der erstellten Seiten für 10,– DM bei den Autoren bekommen. Die Adresse ist beim DMV-Verlag zu erfragen.

Listings

DATABOX

```
;******************************************************
;*                   Turtle.asm                       *
;*     Programm zum Erzeugen von Turtle-Grafiken      *
;*            by M. Cordes & N. Nebel                 *
;*              (c) 1990 AMIGA DOS                    *
;*          Sprache: DevPac Assembler                 *
;******************************************************
PFix    =   -30
PFlt    =   -36
PAdd    =   -66
PSub    =   -72
PMul    =   -78
PDiv    =   -84
PSin    =   -36
PAsin   =   -114
PCos    =   -42
Move    =   -240
Draw    =   -246
SetAPen =   -342
AllocMem    =   -198
FreeMem     =   -210
;******************************************************
    push    macro
    movem.l a0/a1,-(sp)
    endm
```
Listing: Turtle-Grafik

```
    pop     macro
    movem.l (sp)+,a0/a1
    endm
    section text,CODE
    xdef    _lt
    xdef    _rt
    xdef    _fd
    xdef    _penup
    xdef    _pendown
    xdef    _setpos
    xdef    _setheading
    xdef    _ClrScr
    xdef    _Clear
    xdef    _Tabelle
    xdef    _pusht
    xdef    _popt
_lt move    heading,d0
    sub     4(sp),d0
    bsr     korr
    move    d0,heading
    rts
_rt move    4(sp),d0
    add     heading,d0
    bsr     korr
    move    d0,heading
    rts
```
Listing: Turtle-Grafik

```
_setheading
        move    4(sp),d0
        bsr     korr
        move    d0,heading
        rts

_penup          clr penflag
        rts

_pendown
        move    #1,penflag
        rts

_setpos         move.l  4(a7),xpos
        move.l  8(a7),xpos+4
        move.l  12(a7),ypos
        move.l  16(a7),ypos+4
        rts

_fd     move.l  4(a7),d0
        move.l  8(a7),d1
        movem.l d2-d7/a2-a6,-(sp)
        move.l  _ieeebase,a6
        move.l  d0,d2
        move.l  d1,d3
        lea     sint,a0
        move    heading,d0
        lsl     #3,d0
        movem.l (a0,d0.w),d0/d1
        push
        jsr     PMul(a6)
        pop
        move.l  d0,xadd
        move.l  d1,xadd+4
        lea     cost,a0
        move    heading,d0
        lsl     #3,d0
        movem.l (a0,d0.w),d0/d1
        jsr     PMul(a6)
        move.l  d0,yadd
        move.l  d1,yadd+4
        move.l  xpos,d2
        move.l  xpos+4,d3
        move.l  xadd,d0
        move.l  xadd+4,d1
        jsr     PAdd(a6)
        move.l  d0,xnew
        move.l  d1,xnew+4
        jsr     PFix(a6)
        move.l  d0,d6
        move.l  d2,d0
        move.l  d3,d1
        jsr     PFix(a6)
        move.l  d0,d4
        move.l  ypos,d0
        move.l  ypos+4,d1
        move.l  yadd,d2
        move.l  yadd+4,d3
        jsr     PSub(a6)
        move.l  d0,ynew
        move.l  d1,ynew+4
        jsr     PFix(a6)
        move.l  d0,d7
```

Listing: Turtle-Grafik

```
        move.l  ypos,d0
        move.l  ypos+4,d1
        jsr     PFix(a6)
        move.l  d0,d5
        tst     penflag
        beq     fd1
        bsr     line

fd1     lea     xnew,a0
        lea     xpos,a1
        move.l  (a0)+,(a1)+
        move.l  (a0)+,(a1)+
        move.l  (a0)+,(a1)+
        move.l  (a0)+,(a1)+
        movem.l (a7)+,d2-d7/a2-a6
        rts

line            sub d6,d4
        beq     vert
        sub     d7,d5
        beq     horiz
        move    d4,d0
        move    d5,d1
        tst     d0
        bpl.s   lp1
        neg     d0
lp1     tst     d1
        bpl.s   lp2
        neg     d1
lp2
        cmp     d0,d1
        beq     diag
        bgt.s   doy

dox     tst     d4
        bpl.s   dx1
        add     d4,d6
        neg     d4
        add     d5,d7
        neg     d5
dx1     swap    d5
        bpl.s   dx2
        neg.l   d5
        divu    d4,d5
        swap    d5
        clr     d5
        swap    d5
        neg.l   d5
        bra.s   dx3
dx2     divu    d4,d5
        swap    d5
        clr     d5
        swap    d5
dx3     swap    d7
xl1     move    d6,d0
        move.l  d7,d1
        swap    d1
        bsr     plot
        add.l   d5,d7
        addq    #1,d6
        dbf     d4,xl1
        rts

```



Listing: Turtle-Grafik

```
doy   tst     d5
      bpl.s   dy1
      add     d4,d6
      neg     d4
      add     d5,d7
      neg     d5
dy1   swap    d4
      bpl.s   dy2
      neg.l   d4
      divu    d5,d4
      swap    d4
      clr     d4
      swap    d4
      neg.l   d4
      bra.s   dy3
dy2   divu    d5,d4
      swap    d4
      clr     d4
      swap    d4
dy3   swap    d6
yl1   move    d7,d1
      move.l  d6,d0
      swap    d0
      bsr     plot
      add.l   d4,d6
      addq    #1,d7
      dbf     d5,yl1
      rts

vert  sub     d7,d5
      bpl.s   ver1
      add     d5,d7
      neg     d5
ver1  move    d7,d1
      move    d6,d0
      bsr     plot
      addq    #1,d7
      dbf     d5,ver1
      rts

horiz tst     d4
      bpl.s   hor1
      add     d4,d6
      neg     d4
hor1  move    d6,d0
      move    d7,d1
      bsr     plot
      addq    #1,d6
      dbf     d4,hor1
      rts

diag  tst     d4
      bpl.s   dia1
      add     d4,d6
      neg     d4
      add     d5,d7
      neg     d5
dia1  tst     d5
      bpl.s   dia2
      moveq   #-1,d5
      bra.s   dia3
dia2  moveq   #1,d5
dia3  move    d6,d0
```
Listing: Turtle-Grafik




```
      move    d7,d1
      bsr     plot
      addq    #1,d6
      add     d5,d7
      dbf     d4,dia3
      rts

plot     move.l  _plane0,a0
      mulu    #80,d1
      move    d0,d2
      lsr     #3,d0
      add     d0,d1
      add.l   d1,a0
      not     d2
      and     #7,d2
      bset    d2,(a0)
      rts

korr     tst d0
ko2   bpl.s   ko1
      add     #360,d0
      bra.s   ko2
ko1   cmp     #360,d0
      blt.s   ko3
      sub     #360,d0
      bra.s   ko1
ko3   rts


_Tabelle
      movem.l a2-a6/d2-d7,-(a7)
      move.l  _ieeebase,a6
      move.l  _ieeetrans,a5
      moveq   #90,d0
      jsr     PFlt(a6)
      move.l  d0,d2
      move.l  d1,d3
      moveq   #1,d0
      push
      jsr     PFlt(a6)
      jsr     PAsin(a5)
      pop
      jsr     PDiv(a6)
      move.l  d0,d2
      move.l  d1,d3
      push
      jsr     PSub(a6)
      pop
      move.l  d0,d4
      move.l  d1,d5
      lea     sint,a0
      lea     cost,a1
      move    #359,d7
ta1   move.l  d4,d0
      move.l  d5,d1
      push
      jsr     PSin(a5)
      pop
      move.l  d0,(a0)+
      move.l  d1,(a0)+
      move.l  d4,d0
      move.l  d5,d1
      push
```
Listing: Turtle-Grafik

```
        jsr     PCos(a5)
        pop
        move.l  d0,(a1)+
        move.l  d1,(a1)+
        move.l  d4,d0
        move.l  d5,d1
        push
        jsr     PAdd(a6)
        pop
        move.l  d0,d4
        move.l  d1,d5
        dbf     d7,ta1
        move.l  4,a6
        move.l  #18000,d0
        moveq   #0,d1
        jsr     AllocMem(a6)
        move.l  d0,tstack
        move.l  d0,stackp
        movem.l (a7)+,a2-a6/d2-d7
        rts

_Clear  move.l  a6,-(a7)
        move.l  4,a6
        move.l  tstack,a1
        move.l  #18000,d0
        jsr     FreeMem(a6)
        move.l  (a7)+,a6
        rts

_pusht  move.l  stackp,a0
        move.l  xpos,(a0)+
        move.l  xpos+4,(a0)+
        move.l  ypos,(a0)+
        move.l  ypos+4,(a0)+
        move    heading,(a0)+
        move.l  a0,stackp
        rts

_popt   move.l  stackp,a0
        move    -(a0),heading
        move.l  -(a0),ypos+4
        move.l  -(a0),ypos
        move.l  -(a0),xpos+4
        move.l  -(a0),xpos
        move.l  a0,stackp
        rts

_ClrScr move.l  _plane0,a0
        move    #5119,d0
cs1     clr.l   (a0)+
        dbf     d0,cs1
        rts


        section xxx,DATA


taus    ds.l    2
heading ds.w    1
xpos    ds.l    2
ypos    ds.l    2
xadd    ds.l    2
yadd    ds.l    2
```

Listing: Turtle-Grafik




```
xnew    ds.l    2
ynew    ds.l    2
penflag ds.w    1
tstack  ds.l    1
stackp  ds.l    1
sint    ds.l    720
cost    ds.l    720

        xref    _ieeebase
        xref    _ieeetrans
        xref    _GfxBase
        xref    _IntuitionBase
        xref    _rp
        xref    _plane0
;******************** Ende von Turtle.asm ************
```

```
/******************************************************/
/*                     Screen.c                       */
/*              by M. Cordes & N. Nebel               */
/*                 (c) 1990 AMIGA DOS                 */
/*               Sprache: Lattice C                   */
/******************************************************/

#include <intuition/intuition.h>

struct Screen *Screen;
struct Window *Window;

struct IntuitionBase *IntuitionBase;

struct GfxBase *GfxBase;
long ieeebase,ieeetrans;

char *OpenLibrary();
struct Screen *OpenScreen();
struct Window *OpenWindow();

long rp,plane0;

iniscrn()
{
  struct NewScreen ns;
  struct NewWindow nw;

  IntuitionBase = (struct IntuitionBase *)
             OpenLibrary("intuition.library",0L);
  GfxBase = (struct GfxBase *) OpenLibrary("graphics.library",0L);
  ieeebase = (long) OpenLibrary("mathieeedoubbas.library",0L);
  ieeetrans = (long) OpenLibrary("mathieeedoubtrans.library",0L);

  ns.LeftEdge     = 0;
  ns.TopEdge      = 0;
  ns.Width      = 640;
  ns.Height     = 512;
  ns.Depth      = 1;
  ns.DetailPen    = -1;
  ns.BlockPen     = -1;
```

Listing: Turtle-Grafik

```
  ns.ViewModes     = LACE|HIRES;
  ns.Type      = WBENCHSCREEN;
  ns.Font        = NULL;
  ns.DefaultTitle = NULL;
  ns.Gadgets       = NULL;
  ns.CustomBitMap = NULL;

  Screen = OpenScreen(&ns);
  plane0 = (long) (Screen->BitMap.Planes[0]);

  nw.LeftEdge   = 0;
  nw.TopEdge    = 0;
  nw.Width  = 640;
  nw.Height = 512;
  nw.DetailPen  = -1;
  nw.BlockPen   = -1;
  nw.Title  = NULL;
  nw.Flags  = BORDERLESS|ACTIVATE;
  nw.IDCMPFlags = NULL;
  nw.Screen = Screen;
  nw.Type   = WBENCHSCREEN;
  nw.FirstGadget= NULL;
  nw.CheckMark  = NULL;
  nw.BitMap = NULL;
  nw.MinWidth   = 640;
  nw.MinHeight  = 512;
  nw.MaxWidth   = 640;
  nw.MaxHeight  = 512;

  Window = OpenWindow(&nw);
  rp = (long) (Window->RPort);

  Tabelle();
}

clrscrn()
{
  CloseWindow(Window);
  CloseScreen(Screen);
  CloseLibrary(IntuitionBase);
  CloseLibrary(GfxBase);
  CloseLibrary(ieeebase);
  CloseLibrary(ieeetrans);
}
/****************** Ende von Screen.c ****************/
```




```
/*******************************************************/
/*                      Farn.c                         */
/*              by M. Cordes & N. Nebel                */
/*                 (c) 1990 AMIGA DOS                  */
/*                 Sprache: Lattice C                  */
/*******************************************************/

#define back(x) fd(-(x))

void rechteck(laenge,breite,tiefe)
double laenge,breite;
int tiefe;
{

  if (tiefe<=1) {
    lt(90); fd(breite/2); rt(90); fd(laenge);
    rt(90); fd(breite); rt(90); fd(laenge);
    rt(90); fd(breite/2); rt(90);
  } else {
    --tiefe;
    fd(laenge*5/24);
    rt(4); rechteck(laenge*19/24,breite*19/24,tiefe);
lt(4);
    back(laenge/12);
    rt(57); rechteck(laenge/3,breite/3,tiefe-4); lt(57
);
    back(laenge/24);
    lt(65); rechteck(laenge/3,breite/3,tiefe-4); rt(65
);
    back(laenge/12);
  }
}

main ()
{
  int tiefe;
  printf("Tiefe ??? "); scanf("%d",&tiefe);
  iniscrn();

  setpos(320.5L,511.5L);
  setheading(0);
  pendown();
  rechteck(500.0L,250.0L,tiefe);

  while (*((char *)0xbfe001) & 64);

  clrscrn();
}
/******************* Ende von Farn.c ****************/
```

Listing: Turtle-Grafik

```
/*******************************************************/
/*                      Dreieck.c                      */
/*              by M. Cordes & N. Nebel                */
/*                 (c) 1990 AMIGA DOS                  */
/*                 Sprache: Lattice C                  */
/*******************************************************/

dreieck(laenge,tiefe)
double laenge;
int tiefe;
{
  if (tiefe!=1)
  {
              lt(120); fd(laenge); dreieck(laenge/2,t
iefe-1);
    fd(laenge); rt(120); fd(laenge); dreieck(laenge/2,t
iefe-1);
    fd(laenge); rt(120); fd(laenge); dreieck(laenge/2,t
iefe-1);
    fd(laenge); lt(120);
  }
}
```

Listing: Turtle-Grafik

```
main ()
{
  int tiefe;
  double laenge=290.0L;
  printf("Tiefe ? "); scanf("%d",&tiefe);
  iniscrn();

  setpos(30.5L,510.5L);
  setheading(90);
  pendown();
  fd(laenge); dreieck(laenge/2,tiefe); fd(laenge);
  lt(120); fd(laenge*2);
  lt(120); fd(laenge*2);
  while (*((char *)0xbfe001) & 64);

  clrscrn();
}
/*************** Ende von Dreieck.c *****************/
```

```
/*******************************************************/
/*                    Busch.c                          */
/*             by M. Cordes, N. Nebel                  */
/*               (c) 1990 AMIGA DOS                    */
/*              Sprache: Lattice C                     */
/*******************************************************/

double laenge;
do_s(tiefe)
int tiefe;
{
  if (tiefe-- > 1)
  {
    pusht();
    rt(54);
    do_g(tiefe);
    popt();
    pusht();
    lt(54);
    do_g(tiefe);
    popt();
    do_t(tiefe);
    do_s(tiefe);
  }
}
do_g(tiefe)
int tiefe;
{
  if (tiefe-- > 1)
  {
    rt(18);
    do_h(tiefe);
    pusht();
    lt(18);
    do_g(tiefe);
    popt();
    do_l(tiefe);
  }
}
do_h(tiefe)
int tiefe;
{
  if (tiefe-- > 1)
  {
    lt(18);
    do_g(tiefe);
    pusht();
    rt(18);
    do_h(tiefe);
    popt();
    do_l(tiefe);
  }
}
do_t(tiefe)
int tiefe;
{
  if (tiefe-- > 1)
  {
    do_t(tiefe);
    do_l(tiefe);
  }
}
do_l(tiefe)
int tiefe;
{
  pusht();
  lt(18);
  fd(laenge*3);
  popt();
  pusht();
  rt(18);
  fd(laenge*3);
  popt();
  fd(laenge);
}

main ()
{
  int tiefe;
  printf("Tiefe ? "); scanf("%d",&tiefe);
  printf("Laenge ? "); scanf("%lf",&laenge);

  iniscrn();

  setpos(320.5L,511.5L);
  setheading(0);
  pendown();

  do_s(tiefe);
  do_l(tiefe);
  fd(laenge*3);

  while (*((char *)0xbfe001) & 64);

  clrscrn();
  Clear();
}
/***************** Ende von Busch.c *****************/
```



Listing: Turtle-Grafik

Andreas Polk

Freie Fahrt für Modula2

Eine Einführung in die Programmiersprache Modula2



Kursfahrplan 

Teil 1: Der Compiler M2Amiga

Teil 2: Die ersten Befehle

Teil 3: Schleifen und Verschachteln

Teil 4: Prozeduren und Variablen

Teil 5: Weitere Datentypen

Teil 6: Zeiger und Datenstrukturen

Teil 7: Modulkonzept

Teil 8: Systemprogrammierung

Nachdem wir uns im letzten Kursteil mit der Einführung in das Modula2-System beschäftigt haben, wollen wir uns in diesem Kursteil den ersten Befehlen widmen. Alles, was Sie brauchen, sind die Grundlagen aus dem ersten Kursteil. Sie sollten mit dem Editor, Compiler und Linker arbeiten und einen Source-Code in ein ausführbares Programm umwandeln können. Ob Sie dabei mit dem CLI oder der Workbench arbeiten, bleibt Ihnen überlassen.

Jedes Modula2-Programm hat einen bestimmten Aufbau. Dieser muß in jedem Programm eingehalten werden.
Schauen wir uns dazu erst einmal Listing 1 an. Wir haben es im letzen Kursteil bereits benutzt, um mit der Arbeitsweise des Compilers vertraut zu werden. Das erste Befehlswort ist »MODULE«.
Das ist der erste Befehl eines jeden Programms. Wie der Name Modula2 schon sagt, ist Modula modulorientiert. Dies heißt, daß man mit Modula2 verschiedene Programme zu einem zusammenfügen kann. So können verschiedene Programmierer mehrere Programmteile unabhängig voneinander programmieren und diese hinterher zu einem großen Programm zusammenfügen. Die einzelnen Programme, die erstellt werden, nennt man Module. Die verschiedenen Module werden zu einem Hauptmodul, dem fertigen Programm zusammengesetzt.

Da das etwas abstrakt klingt, möchte ich hier ein alltägliches Beispiel aufführen. Nehmen wir an, Sie möchten mit zwei Freunden einen Funktionsplotter programmieren. Da es nicht sehr zweckmäßig ist, immer zu dritt vor dem Rechner zu hocken, sollte man die Programmierarbeit aufteilen. Einer erledigt die Programmierung des Parsers (das ist der Teil, der die eingegebene Formel berechnet), einer kümmert sich um die Oberfläche wie Menüs, File-Select-Box etc., und einer schreibt die Routinen, um den Graphen auf dem Bildschirm auszugeben. Jeder schreibt also einen Programmteil, ein Modul, das unabhängig von den anderen ist. Sind alle Module fertig, müssen sie nur noch zu einem Hauptmodul, dem fertigen Programm, zusammengefügt werden. Da die Programmodule auch miteinander kommunizieren müssen, muß festgelegt werden, wie Daten von einem Modul an ein anderes übergeben werden. Das geschieht über die Schnittstelle. Beispielsweise muß eine Schnittstelle zwischen Parser und dem Plotmodul definiert werden, da der Plotter ja wissen muß, welche Kurve er zu zeichnen hat. Soviel zum Modulprinzip.

Wir werden in späteren Kursteilen noch darauf eingehen, wie man verschiedene Module programmiert. Das Befehlswort »MODULE« gibt also an, daß es sich hier um ein Programmodul handelt, also ein ablauffähiges Programm. Hinter dem Befehlswort folgt der Name des Moduls, hier »HelloWorld«.

Die Dateien, die der Compiler und der Linker erzeugen, bekommen alle den hier angegebenen Namen plus die entsprechende Endung für die Kennzeichnung der Datei. Nach dem Namen des Moduls folgt ein Semikolon.

Das Semikolon spielt bei Modula2 eine wichtige Rolle. Es kennzeichnet das Ende eines Befehls. Nach jedem Befehl muß also (bis auf einige Ausnahmen) ein Semikolon folgen. Nach der Leerzeile, die nur der Übersicht dient, folgt der Befehl »FROM ... IMPORT ....«. Modula2 bietet nur einige wenige Grundbefehle.

Die meisten Befehle werden in Modulen angeboten. So gibt es Module, die für Ein- und Ausgaben, Grafik und so weiter zuständig sind. Die Module enthalten Befehle, die diesen Zwecken dienen. Wollen Sie einen Befehl aus solch einem Modul benutzen, so müssen Sie den Befehl aus diesem Modul importieren, das heißt, dem Programm bekanntmachen, so wie man beispielsweise der Mutter seine Freunde vorstellt. Das geschieht über

den Befehl »FROM Modulname »IMPORT Befehle«.
Wir wollen in diesem Programm einen Befehl zur Ausgabe eines Textes benutzen, der sich in dem Modul »InOut« befindet. Die Befehle heißen »WriteString« und »WriteLn«. So lautet die Zeile »FROM InOut IMPORT WriteString, WriteLn;«. Das Modul heißt »InOut« und die gebrauchten Befehle heißen »WriteString« und »WriteLn«. Sind wie hier mehrere Befehle zu gebrauchen, so müssen diese durch Kommata voneinander getrennt werden. Zum Schluß folgt wieder ein Semikolon.
Natürlich können auch mehrere Befehle aus verschiedenen Modulen benutzt werden. In diesem Fall folgen halt mehrere »FROM ... IMPORT ...«-Anweisungen hintereinander (siehe Kursteil 1, Listing 2).
Jetzt haben wir zwar schon zwei Befehle kennengelert, bis jetzt ist aber noch nichts passiert. Wir haben erst einige Vorbereitungen getroffen. Nun folgen die Programmanweisungen, also die auszuführenden Befehle. Damit der Compiler weiß, daß die Vorbereitungen abgeschlossen sind und nun das eigentliche Programm folgt, wird der Befehl »BEGIN« eingegeben. Dahinter steht kein Semikolon. Alles, was nun an Befehlen kommt, wird später beim Programmablauf ausgeführt. Hier steht zuerst der Befehl »WriteString« und dann »WriteLn«. Auf die Befehle selber werde ich gleich eingehen. Am Ende des Programms steht der Befehl »END«, gefolgt von dem Modulnamen. Dieser Befehl teilt dem Compiler mit, daß das Programm hier zu Ende ist. Durch die Angabe des Programmnamens weiß der Compiler, welches Modul, also hier das Programm selber, beendet wird. Abschließend folgt ein Punkt.
Die Angabe des Programmnamens erscheint hier vorerst überflüssig. Das wäre sie auch, wenn es nicht möglich wäre, mehrere Module zusammenzufassen. Dadurch erreicht man eine bessere Übersichtlichkeit.
In der Abbildung 1 sehen Sie noch einmal den schematischen Aufbau eines Programms. Zuerst steht immer das Befehlswort »MODULE«, gefolgt vom Modulnamen, dann die Importanweisungen, die sogenannte Importliste. Anschließend folgt das eigentliche Programm, das

```
Schematischer Aufbau eines Moduls

MODULE Name
FROM ...   IMPORT....;
.
.
.
.
VAR Variable: Typ;
    .
    .
    .
BEGIN
    Programm
END Name
```

Der Aufbau eines Modula-Moduls

durch ein »BEGIN« eingeleitet wird und mit »END Modulname« und dem anschließenden Punkt beendet wird. Compilieren und starten Sie das Programm, wird der Text "HelloWorld" ausgegeben und der Cursor an den Anfang der nächsten Zeile gesetzt.
Schauen wir uns dazu die beiden Befehle, die dies bewerkstelligen, etwas genauer an. Sie stehen innerhalb von »BEGIN« und »END«, also sind es die Befehle »WriteString« und »WriteLn«.
Der Befehl »WriteString« dient zur Ausgabe einer Zeichenkette auf den Bildschirm. Eine Zeichenkette nennt man übrigens String, und so wird auch der Name dieses Befehls erklärt. Der Text, der ausgegeben werden soll, befindet sich innerhalb der Klammer in Anführungsstrichen. Die Anführungsstriche kennzeichnen diesen Text als String. Würden sie nicht mit angegeben, würde der Text als Variablenname interpretiert. Auf Variablen kommen wir gleich noch zu sprechen. Der Befehl »WriteString« gibt also eine Zeichenkette auf dem Bildschirm aus, in diesem Fall den Text "HelloWorld".
Danach befindet sich der Cursor beim Programmablauf hinter diesem Text. Er soll nun an den Anfang der nächsten Zeile transportiert werden. Das geschieht mit dem nächsten Befehl, »WriteLn«. Er bekommt keine Parameter, denn er setzt lediglich den Cursor auf den Anfang der nächsten Zeile. Danach kommt der Rechner zum END-Befehl und beendet das Programm. Der Text wurde auf dem Bildschirm ausgegeben und der Cursor an den Anfang der nächsten Zeile gesetzt.
Ich möchte an dieser Stelle noch einmal auf das Modulkonzept zurückkommen. Wie bereits gesagt, ist der Befehl »WriteString« ein Teil des Moduls »InOut«. Dieser Befehl dient zur Ausgabe eines Strings. Natürlich muß der Rechner auch wissen, welcher Text, genauer gesagt, welcher String ausgegeben werden soll. Das Hauptprogramm muß also dem Befehl aus dem Modul »InOut« Informationen übergeben, hier den String.
Damit der Befehl des Moduls weiß, wie er mit den übergebenen Daten umgehen muß, wurde vorher festgelegt, wie die Daten übergeben werden. Das geschieht über die bereits erwähnte Schnittstelle. Sie legt fest, welche Daten in welcher Form übergeben werden müssen. Der Anwender des Befehls muß sich lediglich an diese Abmachung halten, und alles wird zufriedenstellend ablaufen. Wie der Befehl letztendlich arbeitet, ist dem Anwender egal, er hat sich nur um den richtigen Aufruf zu kümmern.
Das dient der Abstraktion eines Programms; das heißt, nur die wichtigen Befehle werden angegeben. Wie die Befehle, also hier die Ausgabe des Strings, genau funktionieren, interessiert nicht. Stellen Sie sich vor, Sie sind ein Manager, der nur die Befehle verteilt und sich um die Koordination kümmert. Er gibt seine Anweisungen weiter, braucht sich aber nicht mehr um die Ausführung selbst zu kümmern. Möchten wir Strings ausgeben, haben wir dafür den Befehl »WriteString« gerade kennengelernt. Was ist aber, wenn wir nur Zahlen ausgeben wollen? Dazu verwenden wir einen anderen Befehl. Welcher benutzt wird, hängt von dem auszugebenden Zahlentyp ab. Bevor wir nun den nächsten Befehl kennenlernen, muß ich Sie leider wieder mit etwas Theorie bombardieren, doch das ist beim Lernen einer Programmiersprache unumgänglich.

Zahl ist nicht gleich Zahl

Für den Computer ist Zahl nicht gleich Zahl. Er kennt verschiedene Zahlenformate, genau wie der Mensch auch. Er unterscheidet auch zwischen Bruchzahlen, natürlichen Zahlen, reellen Zahlen... Fangen wir mit der gröbsten Unterscheidung an, zwischen einer Kommazahl und einer ganzen Zahl. Eine ganze Zahl ist eine Zahl ohne Nachkommastellen, beispielsweise 1, 10 oder 100. Eine Kommazahl hat auch Nachkommastellen, wie beispielweise 1.32, -4.5 oder Pi.
Eine ganze Zahl bezeichnet man mit dem Datentyp »INTEGER«. Man spricht dann von einer Integerzahl. Ihr Wertebereich ist das Intervall von -32768 bis 32767. Eine Integerzahl kann also Zahlen zwischen -32768 und 32767 annehmen. Die einzige Voraussetzung ist, daß es sich um eine ganze Zahl handelt. 5.3 befindet sich also nicht in dem Wertebereich einer Integerzahl. Hierbei handelt es sich um eine Kommazahl. Sie erhält den Datentyp »REAL« (reelle Zahlen). Der Wertebereich einer Realzahl umfaßt -1.0*10↑38 bis 1.0*10↑38. Wie Sie sehen, sind hier Kommazahlen zugelassen. Die Genauigkeit beträgt 10↑-6, also sechs Nachkommastellen.
Dies sind die zwei gängigsten Zahlenformate, sozusagen die Grundtypen. Sie wurden in Modula2 noch um einen weiteren Typ erweitert, die sogenannte Kardinalzahl vom Typ »CARDINAL«. Eine Kardinalzahl kann jede natürliche Zahl annehmen, wie der Typ »INTEGER«, allerdings keine negativen Zahlen. Ihre Wertemenge ist auf 0 bis 65535 be-

schränkt. Sie werden sich vielleicht fragen, warum die Grenzen der Wertemengen so unhandliche Zahlen sind. Das liegt daran, daß eine Zahl nicht als Zahl im Speicher abgelegt wird, sondern erst umgewandelt werden muß. Wie dies funktioniert, ist hier nicht wichtig. Warum verschiedene Zahlentypen? Durch die Differenzierung kann der Compiler die Zahlen optimaler handhaben. So braucht eine Integer- und Kardinalzahl im Speicher beispielsweise nur 2 Byte, während eine Realzahl 4 Byte braucht. Die angegebenen Wertebereiche reichen für viele, aber nicht alle Anwendungen aus. Manchmal ist ein größerer Wertebereich nötig, manchmal sind die einfachen Realzahlen mit sechs Nachkommastellen nicht genau genug.

Möchte man also größere Zahlen darstellen, braucht man neue Typen, oder man ändert die alten einfach ab. Hierzu bietet Modula2 noch jeweils eine Erweiterung zu jedem Typ an, die Longzahlen. Sie werden genauso gehandhabt wie ihre "kleinen" Äqivalente, haben allerdings andere Wertebereiche. So besitzt der Typ »LONGCARD« einen Wertebereich von 0 bis 4294967295, also mehr als 4 Milliarden. Der Typ »LONGREAL« besitzt eine Wertemenge von -1.0*10↑308 bis 1.0*10↑308 bei einer Genauigkeit von 15 Nachkommastellen und »LONGINT« (der große Bruder von INTEGER) besitzt einen Wertebereich von -2147483648 bis 2147483647. Diese Bereiche sollten ausreichen.

Der Nachteil dieser Typen ist, daß Sie doppelt soviel Speicherplatz verbrauchen. Eine Auflistung der verschiedenen Variablentypen können Sie der Tabelle 1 entnehmen. Wie Sie gesehen haben, gibt es verschiedene Datentypen. Wir müssen genau beachten, welchen Zahlentyp wir ausgeben wollen.

Für jeden Datentyp gibt es einen eigenen Ausgabebefehl. Für die Ausgabe einer Integerzahl müssen wir den Befehl »WriteInt« benutzen, für die Ausgabe einer Kardinalzahl den Befehl »WriteCard«. Sie haben beide die gleiche Syntax, nämlich »WriteInt(Zahl, rechter Rand) WriteCard(Zahl, rechter Rand)« Zahl ist hier die jeweilige Zahl, die ausgegeben werden soll. Diese muß natürlich vom entsprechenden Typ sein. So können Sie mit dem Befehl »WriteInt« keine Kardinalzahl ausgeben beziehungsweise umgekehrt.

Der Wert "rechter Rand" gibt an, wie breit die Ausgabe der Zahl mindestens sein soll. So ist eine formatierte Ausgabe möglich. Geben Sie hier sechs an, so wird eine Zahl, die mehr als sechs Ziffern lang ist, linksbündig ausgegeben. Hat Sie nur fünf oder weniger Ziffern, wird sie rechtsbündig zum angegebenen Wert ausgegeben.

Hierzu einige Beispiele:

Befehl	Ausgabe
WriteInt(30000,5)	30000
WriteInt(10000,4)	10000
WriteInt(1000,6)	1000
WriteInt(1, 10)	1

Möchten Sie einen Longwert der beiden Zahlentypen ausgeben, so brauchen Sie keinen neuen Befehl zu benutzen. Diese beiden Ausgabebefehle können auch auf die entsprechenden Longwerte angewendet werden. So ist folgender Befehl durchaus erlaubt: »WriteCard(10000000,10);«. Beachten Sie, daß die Zahl 10000000 keine Kardinalzahl mehr ist, sondern eine Longcardzahl.

Diese beiden Befehle finden Sie übrigens auch in dem Modul »InOut«. Möchten Sie sie also in Ihren Programmen verwenden, so müssen Sie die Befehle erst importieren, also dem Compiler bekannt machen. Das geschieht durch folgenden Befehl im Implementationsteil: »FROM InOut IMPORT WriteCard, WriteInt;«.

Was ist, wenn wir nun eine Realzahl ausgeben lassen möchten? Dies geschieht über den Befehl »WriteReal«. Er befindet sich in einem anderen Modul namens »RealInOut«. Wie der Name dieses Moduls schon sagt, ist es nur für die Ein- und Ausgabe von Realzahlen zuständig. Es besitzt auch nur zwei Befehle. Einer lautet »WriteReal« und bewältigt genau das, was wir brauchen. Er gibt eine Realzahl auf dem Bildschirm aus. Seine Syntax lautet: »WriteReal(Zahl, rechter Rand, Dezimalstellen);«. Zahl ist hierbei wieder die auszugebende Realzahl. Der rechte Rand gibt genau wie bei den Integer- und Kardinalzahlen an, wo sich der rechte Rand der Ausgabe befindet. Hat die auszugebende Zahl weniger Ziffern (inklusive Punkt) als beim rechten Rand angegeben, so werden zuerst so viele Leerzeichen ausgegeben, daß die Zahl zum rechten Rand hin rechtsbündig ausgegeben wird. Ist der rechte Rand negativ, so wird die Zahl linksbündig ausgegeben. Mit der Zahl der Dezimalstellen geben Sie an, wie viele Dezimalstellen maximal ausgegeben werden sollen.

Hierzu wieder einige Beispiele.

Befehl	Ausgabe
WriteReal(100.23,4,4)	100
WriteReal(100.23,6,4)	100.23
WriteReal(100.23,8,1)	100.2
WriteReal(100.23,-8,1)	100.2000

Möchten Sie mit Longrealzahlen arbeiten, so bedienen Sie sich der Routinen des Moduls »LongRealInOut«. Sie heißen genauso wie die Routinen des Moduls »RealInOut« und werden genauso bedient, nur müssen Longrealzahlen übergeben werden.

In Listing 2 habe ich Ihnen ein Programm aufgelistet, mit dem Sie dies ausprobieren können. Am Ende dieses Kursteils sind Sie soweit, daß Sie dieses Programm schon verstehen können. Denjenigen, die mit der PD-Version des Compilers arbeiten, muß ich leider mitteilen, daß die beiden Module »RealInOut« und »LongRealInOut« sich nicht auf der Diskette befinden. Sie können die Programme, in denen diese

```
Die Module InOut, RealInOut, LongRealInOut und Terminal
Dieser Referenzteil enthaelt die wichtigsten Ausgabemodule

Das Modul InOut

Es gelten folgende Deklarationen:
VAR Zeichen          : CHAR;
    String, Endung   : ARRAY OF CHAR;
    IntZahl          : INTEGER;
    LongIntZahl      : LONGINT;
    CardZahl         : CARDINAL;
    LongCardZahl     : LONGCARD;
    Block            : ARRAY OF BYTE;

Hier die Prozeduren:
Read(Zeichen);
ReadString(String);
ReadInt(IntZahl);
ReadLongInt(LongIntZahl);
ReadCard(CardZahl);
ReadLongCard(LongCardZahl);
ReadBytes(Block);
Write(Zeichen);
WriteLn;
WriteString(String);
WriteInt(IntZahl) oder WriteInt(LongIntZahl);
WriteCard(CardZahl) oder WriteCard(LongCardZahl);
WriteOct(LongIntZahl, Breite);
WriteHex(LongIntZahl, Breite);
WriteBytes(Block);
OpenInput(Endung);
OpenOutput(Endung);
CloseInput;
CloseOutput;

Das Modul Terminal

Deklarationsteil:

VAR Zeichen : CHAR;
    String : ARRAY OF CHAR;
    Laenge : INTEGER;
Prozeduren:
Read(Zeichen);
ReadLn(String, Laenge);
BusyRead(Zeichen);
Write(Zeichen);
WriteLn;
WriteString(String);
Anmerkung: Die meisten Befehle dieses Moduls arbeiten wie die
Prozeduren aus InOut, nur beziehen Sie sich immer auf die
Tastatur, whrend sich die Befehle aus InOut auch auf Dateien
beziehen knnen.

Das Modul RealInOut

Deklarationsteil:

VAR RealZahl           : REAL;
    Breite, Nachkomma : INTEGER;
Prozeduren:
ReadReal(RealZahl);
WriteReal(RealZahl, Breite, Nachkomma);

Das Modul LongRealInOut

Deklarationsteil:
VAR LongRealZahl       : LONGREAL;
    Breite, Nachkomma : INTEGER;
Prozeduren:
ReadReal(RealZahl);
WriteReal(RealZahl, Breite, Nachkomma);
```

Tabelle 1 zeigt die besprochenen Module

Module gebraucht werden, also nicht compilieren. Sie finden aber sowohl die Source-Codes als auch die ausführbaren Programme natürlich wie immer auf der Databox.

An dieser Stelle wollen wir einen kleinen Einschnitt machen. Was haben wir bis jetzt kennengelernt? Sie wissen, wie ein Programm strukturiert ist, und haben das Modulkonzept kennengelernt. Verschiedene Datentypen, wie Strings, »INTEGER«, »CARDINAL« und »REAL«, kennen Sie ebenfalls. Ausgeben können wir solche Zahlen mit den Befehlen »WriteInt«, »WriteCard«, »WriteReal« und »WriteString« auch schon. Auch den Cursor können wir mit »WriteLn« auf den Anfang der nächsten Zeile setzen.

Rechnen mit Modula2

Die wichtigsten Rechenarten sind die Addition, Subtraktion, Multiplikation und Division. Wie allgemein üblich, wird die Addition durch das Pluszeichen zwischen beiden Summanden kenntlich gemacht, die Subtraktion durch ein Minuszeichen. Um zwei Zahlen zu addieren und auszugeben, brauchen Sie lediglich »WriteInt(5+3, 4);« zu schreiben, und das Ergebnis wird auf dem Bildschirm ausgegeben. Die Subtraktion funktioniert analog dazu. Auch die Multiplikation funktioniert genauso. Hier ist der Operator das Malzeichen.

Bei der Division werden wir allerdings vor ein kleines Problem gestellt. Was passiert, wenn das Ergebnis nicht ganzzahlig ist? Eine einfach Division 5/3 darf nicht angewendet werden, denn das Ergebnis ist nicht ganzzahlig. Also hat man sich einen anderen Operator einfallen lassen, den Operator »DIV«. Er dividiert zwei Zahlen und schneidet beim Ergebnis einfach die Nachkommastellen ab. Gerundet wird nicht. So ergibt (5 DIV 3) als Ergebnis 1 und nicht 2.

Brauchen Sie den Rest einer Division, so gibt es einen weiteren Operator, nämlich »MOD«. Er liefert als ganze Zahl den Rest einer Division. (5 MOD 3) ergibt als Ergebnis 2, da 5/3 1 Rest 2 ist. Ein Beispielprogramm zum Rechnen mit ganzen Zahlen sehen Sie in Listing 3.

Für Kardinalzahlen gelten genau die gleichen Operatoren, weshalb ich nicht näher auf die Kardinalzahlen eingehe. Sie müssen natürlich beachten, daß Sie den Wertebereich der Kardinalzahlen nicht verlassen. So ist die Anweisung »WriteCard(5-6, 4);« nicht zulässig, da das Ergebnis -1 lautet und sich nicht mehr im Wertebereich von Kardinalzahlen befindet. Wenn der Compiler das nicht schon vor dem Compilieren beanstandet, wird das Programm spätestens beim Ablaufen mit einem Runtime-Error abbrechen.

Wenn wir mit Realzahlen rechnen wollen, können wir genauso die Operatoren (+), (-) und (*) nehmen. Für die Division gibt es das gängige Geteiltzeichen (/). Um also zwei Zahlen zu dividieren, müssen wir lediglich »WriteReal(5.0/3.0, 5, 3);« schreiben. Das Ergebnis lautet in diesem Fall 1,666.

In Listing 4 sehen Sie auch hier ein Beispiel für das Rechnen mit Realzahlen.

Variablen

Wir wollen uns nun einem weiteren wichtigen Thema zuwenden, nämlich den Variablen. In den bisherigen Programmen wurden Zahlen immer direkt angegeben. Meistens ist es aber viel sinnvoller, den Zahlen einen Namen zu geben. Diese Namen nennt man Variablen. Eine Variable ist also ein Platzhalter für einen veränderbaren Wert. Dies muß nicht unbedingt eine Zahl, sondern kann auch ein String oder ein anderer Datentyp, den wir gleich noch kennenlernen werden, sein. Wer sich ein bißchen mit Mathematik auskennt, wird mit Variablen schnell zurechtkommen. Für alle diejenigen, die mit diesem Begriff nichts anfangen können, hier ein kleines Anwendungsbeispiel.

Nehmen wir an, wir wollen ein Programm schreiben, das den Benzinverbrauch auf 100 Kilometer berechnet. Der Anwender muß eingeben, wieviel Kilometer er gefahren ist und wieviel Benzin er verbraucht hat. Diese Zahlen werden Variablen zugeordnet, mit denen nun gerechnet werden muß, denn wie sollte der Rechner sich sonst eine Zahl merken? So weist man der Variablen "Kilometer" die gefahrenen Kilometer zu und der Variablen "Benzin" die verbrauchten Liter. Wir werden dieses Programm am Ende dieses Kurses noch erstellen.

Variablen können nicht einfach so verwendet werden. Der Programmierer muß sich erst klarmachen, welche Art von Daten in der Variablen abgelegt werden sollen und wie die Variablen heißen. Wir haben schon Variablentypen kennengelernt, nämlich »INTEGER«, »CARDINAL« und »REAL« und deren lange Typen. Wie machen wir dem Compiler nun klar, welche Variablen wir benutzen wollen und wie sie heißen? Dies geschieht im Deklarationsteil unseres Programms. Er wird eingeleitet durch den Befehl »VAR«. Anschließend folgen die Variablennamen gefolgt von einem Doppelpunkt und der Typenangabe. Der Deklarationsteil folgt direkt nach der Importliste. Somit wird unsere Programmstruktur um einen weiteren Programmteil erweitert (siehe Bild 2).

Auch hier ein Beispiel zur Veranschaulichung: Wir wollen zwei Variablen deklarieren, die eine soll eine Realzahl aufnehmen und die andere eine Kardinalzahl. Der Deklarationsteil sieht dann folgendermaßen aus: »VAR Realzahl : REAL; Kardinalzahl : CARDINAL;«. Zuerst wird das Befehlswort »VAR« angegeben. Dann folgt der Name der ersten Variablen. Damit der Rechner weiß, um welchen Variablentyp es sich hier handelt, wird hinter dem Doppelpunkt der Datentyp angegeben. Anschließend folgt – wie immer – ein Semikolon. Sollen weitere Variablen deklariert werden, so können Sie nach diesem Schema einfach fortfahren, wie ich dies hier für eine weitere Kardinalvariable getan habe.

Beachten Sie zwei Dinge: Das Befehlswort »VAR« darf nur einmal angegeben werden, und zwar am Anfang der Deklaration. Die Leerzeichen zwischen dem Variablennamen und dem Doppelpunkt beziehungsweise zwischen Doppelpunkt und Datentyp sind optional, das heißt sie können auch weggelassen werden. Sie dienen lediglich der Übersicht.

Variablenzuweisung

Nach einer Deklaration ist die Variable zwar bekannt, sie besitzt aber noch keinen Wert. Die Wertzuweisung müssen wir im Hauptprogramm vornehmen. Eine Wertzuweisung bei der Deklaration, so wie es die Sprache C erlaubt, ist nicht möglich. Eine Wertzuweisung sieht in Modula2 folgendermaßen aus:

```
Realzahl:=3.5;
Kardinalzahl:=5;
```

Zuerst muß der Variablenname angegeben werden. Dieser Variablen wird nun ein Wert zugeteilt. Dies wird durch den Doppelpunkt mit Gleichheitszeichen kenntlich gemacht. Immer, wenn Sie ihn sehen, wird einer Variablen ein Wert zugewiesen. Anschließend folgt der Wert, den die Variable erhalten soll, hier wird der Variablen »Realzahl« der Wert 3.5 und der Kardinalzahl der Wert 5 zugewiesen.

Bei der Wertzuweisung müssen Sie einige Dinge beachten. Es kommt nämlich immer darauf an, welchen Variablentyp die Variable hat. So muß einer Realvariablen immer eine Kommazahl zugewiesen werden, auch wenn sie keine Nachkommastellen hat. Eine Zuweisung wie »Realzahl: = 5;« geht nicht. Richtig muß es lauten: »Realzahl: = 5.0;«. Dies bewirkt genau das gleiche, erzeugt aber keinen Fehler.

Kardinal- und Integerzahlen dürfen nur ganze Zahlen zugewiesen werden. Eine Zuweisung wie »Kardinalzahl: = 5.0;« ist unzulänglich. Sie lautet richtig: »Kardinalzahl: = 5;« Natürlich können wir auch mit Variablen rechnen. Hier liegt eine der Hauptanwendungen von Variablen. Es gelten die gleichen Rechenbedingungen wie beim direkten Rechnen mit Zahlen.

Der Vorteil beim Rechnen mit Variablen liegt darin, daß das Ergebnis nicht immer direkt

Typ und Wertebereich

»INTEGER« [-32.768;32.767]

»LONGINT« [-2.147.483.648;2.147.483.647]

»CARDINAL« [0;65535]

»LONGCARD« [0;4.294.967.295]

»REAL« [-1.0*10↑38;1.0*10↑38]

»LONGREAL« [-1.0*10↑308;1.0*10↑308]

ausgegeben werden muß, sondern in einer anderen Variablen zwischengespeichert werden kann. Ein Beispiel zum Rechnen mit Variablen sehen Sie in Listing 5. Schauen wir uns einmal den Deklarationsteil genauer an. »VAR Zahl1, Zahl2, Ergebnis : REAL;«, hier haben wir etwas Neues. In einer Zeile werden gleich drei Variablen deklariert. Dies ist immer dann möglich, wenn die Variablen den gleichen Datentyp besitzen sollen. Die verschiedenen Variablennamen müssen dann einfach durch Kommata voneinander getrennt werden. Ansonsten finden Sie in dem Listing nichts Neues.

Im Programmteil werden den beiden Variablen »Zahl1« und »Zahl2« zwei Werte und dann der Variablen »Ergebnis« verschiedene Rechenergebnisse zugewiesen. Anschließend wird das Ergebnis ausgegeben. Dies wiederholt sich mit allen vier Grundrechenarten. Wir hätten das Programm auch etwas abkürzen können. So hätten wir die Variable Ergebnis einsparen können und das Ergebnis einfach mit »WriteReal(Zahl1 + Zahl2, 10, 5);« ausgegeben können.

Sie sehen, auch mit Variablen kann man rechnen und das Ergebnis direkt ausgeben. Um mit Integer- oder Kardinalzahlen zu rechnen, gilt das gleiche wie für Realzahlen. Beachten Sie aber, daß Sie für die Ganzzahldivision den Operanden »DIV« benutzen und für die Restdivision den Operanden »MOD«.

Weitere Datentypen

Die Typen »REAL«, »INTEGER« und »CARDINAL« sind die einzigen Datentypen, die Modula2 kennt. Ich möchte Ihnen noch einige weitere Elementartypen erklären. Da hätten wir zum ersten den Datentyp »CHAR«. »CHAR« ist die Abkürzung für Character (= Zeichen). In einer Char-Variablen können wir also ein Zeichen ablegen, etwa einen Buchstaben oder ein Sonderzeichen.

Im Deklarationsteil müssen wir einer Variablen den Datentyp »CHAR« zuordnen. Das sieht folgendermaßen aus: »VAR Zeichen : CHAR;«. Wir können nun im Hauptprogramm der Variablen Zeichen ein Zeichen zuweisen. Das geschieht mit folgender Anweisung: »Zeichen: = "A";«. Beachten Sie, daß das zugewiesene Zeichen in Anführungsstrichen stehen muß.

Wie Sie vielleicht wissen, wird jedem Zeichen intern eine Zahl zwischen 0 und 255 zugewiesen. Welche Zahl welchen Wert besitzt, können Sie in einer ASCII-Tabelle nachlesen. Ich erzähle Ihnen das hier, da Modula2 dies durch zwei Befehle unterstützt. Nehmen wir einmal an, eine Char-Variable soll ein Zeichen zugewiesen bekommen, das Sie nicht auf der Tastatur finden, dessen ASCII-Code Sie aber aus einer Tabelle kennen. Dann können Sie der Char-Variablen dieses Zeichen zuweisen, indem Sie die Funktion »VAL« nutzen. Sie hat folgende Syntax: »VAL(Datentyp, Wert);«.

Hier handelt es sich um den Datentyp »CHAR«. Wir müssen nun aus einer ASCII-Tabelle einen Wert heraussuchen. Beispielsweise hat das große A den Wert 65. So können wir der Variablen »Zeichen« das große A auch durch folgende Anweisung zuweisen: »Zeichen: = VAL(CHAR, 65);«. Diese Anweisung bewirkt genau das gleiche wie die oben genannte. Interessant wird dies erst, wenn man Codes von nicht druckbaren Zeichen angibt. Diese Zeichen nennt man Steuerzeichen. Ein solches ist der Code 7. Wird dieses Steuerzeichen ausgegeben, so blinkt der Bildschirm einmal auf. Ein Programm, welches genau dies macht, sehen Sie in Listing 6.

Sie sollten in der Lage sein, das Programm vollkommen zu verstehen. Das einzige Neue ist hier die Ausgabeprozedur »Write«. Sie ist für die Ausgabe eines Zeichens zuständig. Ansonsten wird Sie genauso benutzt wie »WriteString«. Beachten Sie, daß die Variable »Zeichen« nicht in Anführungsstrichen stehen darf, denn sonst würde der Text »Zeichen« nicht mehr als Variable, sondern als Text, also als String, interpretiert. Das ergäbe eine Fehlermeldung, denn »Write« kann nur einzelne Zeichen ausgeben, nicht ganze Strings. Möchten Sie mit »Write« ein Zeichen direkt ausgeben, so muß dieses aber in Anführungsstrichen stehen. Der Befehl »Write("A");« würde auf dem Bildschirm ein großes A ausgeben.

Um Ihnen die verschiedenen Möglichkeiten zur Ausgabe von Zeichen noch einmal klarzumachen, hier drei verschiedene Möglichkeiten, dies zu tun: »Write("A");«, «Write(VAL (CHAR, 65));«, «Write(Zeichen);«, wobei im letzten Beispiel die Variable »Zeichen« das große A enthalten muß.

Merken Sie sich: Alles, was in Anführungsstrichen steht, wird als String interpretiert. Befehle dürfen in der Regel nicht in Anführungsstrichen stehen!

Möchten Sie von einem Zeichen den ASCII-Code erfahren, so dient dazu die Funktion »ORD«. Sie ist praktisch das Gegenstück zu »VAL«. Ihre Syntax lautet: »Kardinalzahl: = ORD(Variable);«.

Variable ist in diesem Falle die Char-Variable. Zurückgegeben wird in der Variablen »Kardinalzahl« die Ordnungszahl des Zeichens. Beachten Sie, daß die Zahl als Kardinalzahl deklariert werden muß, sonst bricht der Compiler mit einer Fehlermeldung ab. Haben Sie vorher der Variablen »Zeichen« durch die Anweisung »Zeichen: = "A";« das Zeichen A zugewiesen, so können Sie sich den zugehörigen Code mit »ZeichenCode: = ORD(Zeichen); WriteCard(Zeichencode);« ausgeben lassen. Beachten Sie, daß Zeichencode als Kardinalvariable deklariert wurde. Kürzer geht dies auch mit einem Befehl: »WriteCard(Ord(Zeichen));«.

Sie sehen, solche Abkürzungen sind erlaubt. Der Befehl »ORD« gibt eine Kardinalzahl zurück, die direkt ausgegeben wird. Eine Abwandlung des Datentypes »CHAR« ist der String. Diesen Datentyp kennt Modula2 nicht. Man muß ihn sich selber bauen. Dies geschieht mit der Array-Anweisung. Ich werde darauf in Teil 5 genauer eingehen, hier nur soviel: Möchten Sie eine Stringvariable benutzen, so müssen Sie sie folgendermaßen deklarieren: »VAR String : ARRAY[0..Groesse] OF CHAR;« wobei »Groesse« die maximale Länge des Strings ist. Beispielsweise könnte eine Deklaration folgendermaßen lauten: »VAR String : ARRAY[0..100] OF CHAR;«. String ist in diesem Fall ein String, der 101 Zeichen aufnehmen kann.

Mit folgender Zuweisung können Sie der Variablen nun einen String zuweisen: »String: = "Hallo, wie geht's?";« Beachten Sie, daß der String in Anführungsstrichen stehen muß. Zur Ausgabe eines Strings kennen wir ja schon den Befehl »WriteString«.

Als letzten Datentyp möchte ich hier den Typ »BOOLEAN« erwähnen. Eine Boolean-Variable kann nur genau zwei Werte annehmen: »TRUE« und »FALSE«, also wahr oder falsch. Das ist dann sinnvoll, wenn nur überprüft wird, ob eine Aussage wahr oder falsch ist. Deklariert wird eine Boolean-Variable folgendermaßen: »VAR Auswertung : BOOLEAN;«. Der Variablen »Auswertung« kann nun der Wert »TRUE« oder »FALSE« durch folgende Zuweisungen zugewiesen werden: »Auswertung: = TRUE; Auswertung: = FALSE;«. Wir werden diesen Variablentyp im nächsten Kursteil sehr oft benutzen. Hier sei er schon einmal der Vollständigkeit halber erwähnt, denn nun haben wir alle Standardtypen besprochen.

Eingabe von Werten

Bisher lief unser Programm immer ab, ohne daß der Benutzer den Ablauf beeinflussen wollte. Möchte er das tun, so muß das Programm mit ihm kommunizieren. Eine einfache Art der Kommunikation ist die Eingabe von Daten. Dazu finden wir im Modul »InOut« einige Standardbefehle. Dies sind die Read-Befehle.

Möchten Sie beispielsweise während des Programmablaufs einer Kardinalvariablen einen Wert zuweisen, so können Sie dazu den Befehl »ReadCard« benutzen. Er hat folgende Syntax: »ReadCard(Variable);«. Der Variablen »Variable«, die vom Typ »Cardinal« sein muß (ist ja logisch), wird der eingegebene Wert zugewiesen. Kommt das Programm zu diesen Befehl, so wird solange gestoppt, bis der Benutzer eine Zahl eingegeben und [Return] gedrückt hat. Dann wird »Variable« die eingegebene Zahl zugeordnet und das Programm fährt weiter fort.

Geben Sie einen Wert an, der nicht zulässig ist, wird der Programmablauf abgebrochen. Deshalb ist es sinnvoller, dem Benutzer mitzuteilen, welchen Wert er eingeben soll. Dies können Sie mit dem Befehl »WriteString« tun. Analog zu dem Befehl »ReadCard« funktionieren die Befehle »ReadLongCard«, »ReadInt« und »ReadLongInt« (siehe Referenz). Natürlich müssen die Variablen hier entsprechenden Typs sein. Beachten Sie, daß

bei der Eingabe von Zahlen zwischen den kurzen und langen Typen unterschieden wird, nicht jedoch bei der Ausgabe! Möchten Sie ein einzelnes Zeichen eingeben, so können Sie dazu den Befehl »Read« benutzen. Er hat die Syntax »Read (Zeichen);«.

Der Variablen »Zeichen« wird dann der eingegebene Buchstabe übergeben. Geben Sie mehr als ein Zeichen ein, so wird das Ende der Zeichenkette einfach abgeschnitten. Möchten Sie einer Realvariablen einen Wert zuweisen, so müssen Sie sich des Befehls des Moduls »RealInOut« bedienen. Der Befehl lautet analog »ReadReal(Realzahl);«.

Auch hier wird der Variablen »Realzahl« der Wert zugewiesen, den der Benutzer eingegeben hat. Achten Sie darauf, das Komma mit einzugeben. Der letzte Befehl, den Sie heute kennenlernen, ist der Befehl »ReadString« aus dem Modul »InOut«. Er weist einer Stringvariablen eine Zeichenkette zu, die der Benutzer eingeben kann. Die Syntax lautet: »ReadString(String);«.

Faßt String weniger Zeichen als eingegeben, so wird der Rest einfach wieder abgeschnitten.

Wir wollen die heute gelernten Befehle nun einmal in die Praxis umsetzen. Realisieren wir das Programm, das aus der Anzahl der gefahrenen Kilometer und dem absoluten Benzinverbrauch den Benzinverbrauch auf 100 Kilometer berechnet.

Was brauchen wir dazu? Zuerst einmal muß der Benutzer die beiden benötigten Daten eingeben.

Ein Beispiel

Diese speichern wir am besten in Realvariablen ab, da das Ergebnis später auf eine Nachkommastelle genau sein soll. Der Benutzer muß also Realzahlen eingeben, dazu kennen wir den Befehl »ReadReal« aus dem Modul »RealInOut«. Das Ergebnis, ebenfalls eine Realzahl, muß hinterher wieder ausgegeben werden, also benutzen wir dazu den Befehl »WriteReal« aus dem gleichen Modul. Da auch noch einige Hilfstexte ausgegeben werden, brauchen wir auch noch die Befehle »WriteString« und »WriteLn« aus dem Modul »InOut«. Sie sollten mit diesen kleinen Hilfen in der Lage sein, das Programm selbständig zu schreiben. Zum Vergleich habe ich es in Listing 7 noch einmal abgedruckt. Hier können Sie Ihre Version mit meiner vergleichen.

So, das war's für diesen Teil. Probieren und spielen Sie mit den Befehlen herum. Wie wäre es mit einem Rechenprogramm, oder einem Programm, welches Zahlen auf verschiedene Weisen formatiert ausgibt (Tip: Die Parameter Rechter-Rand und Größe der Prozedur »WriteReal« verändern! (siehe Listing 4))

(cd)

Listings

```
MODULE HelloWorld;

FROM InOut IMPORT WriteString, WriteLn;

BEGIN
  WriteString("Hello World!");
  WriteLn;
END HelloWorld.
```

```
MODULE RealDemo;

FROM RealInOut IMPORT WriteReal, ReadReal;
FROM InOut IMPORT WriteString, WriteLn, ReadInt;

VAR Zahl : REAL;
    m, n : INTEGER;

BEGIN
  WriteString("Zahl: ");
  ReadReal(Zahl);
  WriteString("Größe des Feldes: ");
  ReadInt(m);
  WriteString("Nachkommastellen: ");
  ReadInt(n);
  WriteReal(Zahl, m, n);
  WriteLn;
END RealDemo.
```

```
MODULE IntDemo;

FROM InOut IMPORT WriteString, WriteLn, WriteInt;

BEGIN
  WriteString("5+3=");
  WriteInt(5+3, 3);
  WriteLn;
  WriteString("5-3=");
  WriteInt(5-3, 3);
  WriteLn;
  WriteString("5*3=");
  WriteInt(5*3, 3);
  WriteLn;
  WriteString("5/3=");
  WriteInt(5/3, 3);
  WriteLn;
END IntDemo.
```

```
MODULE RealDemo2;

FROM RealInOut IMPORT WriteReal;
FROM InOut IMPORT WriteString, WriteLn;

BEGIN
  WriteString("5.0+3.0=");
  WriteReal(5.0+3.0, 6, 3);
  WriteLn;
  WriteString("5.0-3.0=");
  WriteReal(5.0-3.0, 6, 3);
  WriteLn;
  WriteString("5.0*3.0=");
  WriteReal(5.0*3.0, 6, 3);
  WriteLn;
  WriteString("5.0/3.0=");
  WriteReal(5.0/3.0, 6, 3);
  WriteLn;
END RealDemo2.
```

Listing: Modula2-Kurs

```
MODULE RechenDemo2;

FROM InOut IMPORT WriteString, WriteLn;
FROM RealInOut IMPORT WriteReal;

VAR Zahl1, Zahl2, Ergebnis : REAL;

BEGIN
  Zahl1:=5.0;
  Zahl2:=3.0;
  WriteString("Zahl1: ");
  WriteReal(Zahl1, 6, 3);
  WriteLn;
  WriteString("Zahl2: ");
  WriteReal(Zahl2, 6, 3);
  WriteLn;
  WriteLn;
  WriteString("Zahl1 + Zahl2: ");
  Ergebnis:=Zahl1+Zahl2;
  WriteReal(Ergebnis, 6, 3);
  WriteLn;
  WriteString("Zahl1 - Zahl2: ");
  Ergebnis:=Zahl1-Zahl2;
  WriteReal(Ergebnis, 6, 3);
  WriteLn;
  WriteString("Zahl1 * Zahl2: ");
  Ergebnis:=Zahl1*Zahl2;
  WriteReal(Ergebnis, 6, 3);
  WriteLn;
  WriteString("Zahl1 / Zahl2: ");
  Ergebnis:=Zahl1/Zahl2;
  WriteReal(Ergebnis, 6, 3);
  WriteLn;
END RechenDemo2.
```

```
MODULE VALDemo;

FROM InOut IMPORT Write, WriteLn;

VAR Zeichen : CHAR;

BEGIN
  Zeichen:=VAL(CHAR, 7);
  Write(Zeichen);
  WriteLn;
END VALDemo.
```

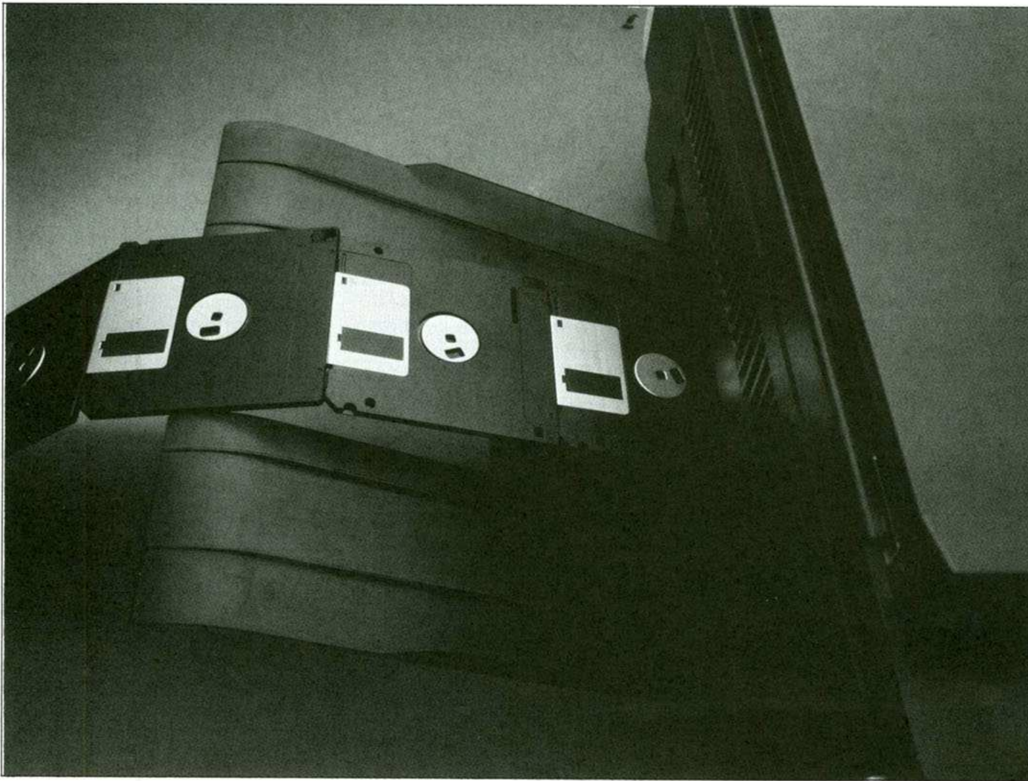
```
MODULE Benzin;

FROM InOut IMPORT WriteString, WriteLn;
FROM RealInOut IMPORT ReadReal, WriteReal;

VAR Benzin, Kilometer : REAL;

BEGIN
  WriteString("Geben Sie die gefahrenen Kilometer ein: ");
  ReadReal(Kilometer);
  WriteString("Geben Sie das verbrauchte Benzin an: ");
  ReadReal(Benzin);
  WriteLn;
  WriteString("Verbrauch auf 100 Kilometern: ");
  WriteReal((Benzin/Kilometer)*100.0, -5, 1);
  WriteLn;
END Benzin.
```

Listing: Modula2-Kurs



Dirk Rönnau

Kopieren, was das Zeug hält

»CLI-Copy« macht's möglich

Beim Diskettenkopieren scheiden sich oft die Geister. Die meisten mögen's umständlich: Raus aus dem CLI, über die Workbench kopiert, wieder rein ins CLI und weitergearbeitet. Warum umständlich, wenn's auch leichter geht.

Hier schafft unser Programm »CLI-Copy« Abhilfe. Es handelt sich dabei, wie der Name schon sagt, um ein Kopierprogramm. Mit ihm lassen sich Amiga-DOS-Disketten ohne Kopierschutz kopieren. Die reine Kopierzeit beträgt 1 min 18 sec. Es ist damit nur 9 Sekunden langsamer als das kommerzielle Programm »X-Copy« im Doscopy-Modus. Zusätzlich lassen sich mit »CLI-Copy« auch Disketten checken und formatieren.

Nach Aufruf von »CLI-Copy« öffnet sich ein kleines Window, das in drei Bereiche unterteilt ist. Der obere Teil des Windows stellt die Source- und Destination-Disk mit ihren jeweils 80 Tracks dar. In der Grundeinstellung sind die Laufwerke DF0: und DF1: voreingestellt. Durch Anklicken dieser Bezeichnungen lassen sich die Laufwerkszuordnungen beliebig ändern.

Der zweite Bereich des Windows ist in vier verschiedene Modi (»COPY«, »CHECK«, »FORMAT«, »NAME«) unterteilt. Der Befehl »COPY« ermöglicht es, ein Backup der jeweiligen Diskette zu machen. Nur in diesem Modus kann man beide Laufwerksbezeichnungen durch Anklicken verändern. »CHECK« untersucht eine Diskette auf eventuelle Lesefehler. »FORMAT« formatiert und initialisiert eine Diskette. Mit dem letztem Modus »NAME« kann man sich den Namen einer Diskette ausgeben lassen. Das ist vor allem sinnvoll, um vor dem Kopieren und Formatieren sicherzustellen, daß man auch die richtige Diskette erwischt hat.

Im dritten Bereich des Windows hat man schließlich die Möglichkeit, die angewählten Modi entweder zu aktivieren (»START PROCESS«) oder zu stoppen (»ABORT PROCESS«). Will man beispielsweise das Formatieren einer Diskette abbrechen, klickt man einfach das Feld »ABORT PROCESS« an und hält die linke Maustaste solange gedrückt, bis das Feld invertiert. Ein Read- beziehungsweise Write-Error wird durch einen kleinen Punkt im betreffenden Track gekennzeichnet. Wird kein Error festgestellt, so wird der gesamte Track in einer neuen Farbe dargestellt. Das waren die wichtigsten Informationen zu diesem Programm. Frisch ans Werk und munter draufloskopiert.

(vb)

Listings

```
;*******************************************************
;*                     CLI-COPY                        *
;*               Diskettenkopierprogramm               *
;*        (c) 1990 Dirk Roennau & AMIGA DOS            *
;*         Sprache: Kuma-Seka Assembler V1.5           *
;*******************************************************

        execbase = 4
        mousebot = $bfe001
        allocmem = -198
        freemem  = -210
        openlib  = -408
        closelib = -414
        setApen  = -342
        setBpen  = -348
        ViewAdd  = -294
        recfill  = -306
        openwb   = -210
        openwin  = -204
        closewin = -72
        text     = -60
```

DATABOX

Listing: CLI-Copy

```

;*************** Buffer reservieren *****************

        move.l    execbase,a6

do_allocate:
        move.l    #512*11*4,d0
        move.l    #$10003,d1
        jsr       allocmem(a6)
        move.l    d0,bufBase

;*************** Libs oeffnen ***********************

        lea       intname,a1
        jsr       openlib(a6)
        move.l    d0,intbase
        lea       graname,a1
        jsr       openlib(a6)
        move.l    d0,grabase

;*************** screen handle setzen ***************

        move.l    intbase,a6
        jsr       openwb(a6)
```

Listing: CLI-Copy

```
        move.l      d0,scrHandle

;*************** window oeffnen *********************

        lea         window_def,a0
        jsr         openwin(a6)
        move.l      d0,winhandle
        move.l      d0,a0
        move.l      50(a0),rastport
        moveq       #2,d0
        bsr         do_Apen

;*************** Fenster einfaerben *****************

        moveq       #2,d0
        moveq       #10,d1
        move        #237,d2
        moveq       #60,d3
        bsr         do_recfill

;*************** Fenster LayOut *********************

        moveq       #0,d0
        bsr         do_Apen
        moveq       #2,d0
        bsr         do_Bpen
        moveq       #0,d7
        bsr         do_Dtxt
        moveq       #112,d7
        bsr         do_Dtxt
        move.l      rastport,a1
        moveq       #84,d1
        move        d1,38(a1)
        bsr         do_line
        move.l      rastport,a1
        moveq       #70,d1
        move        d1,38(a1)
        bsr         do_line

;*************** Message Box Line *******************

        move.l      rastport,a1
        moveq       #60,d1
        move        d1,38(a1)
        bsr         do_line

;*************** Trennbox clear/save ****************

        moveq       #119,d0
        moveq       #85,d1
        moveq       #121,d2
        moveq       #98,d3
        bsr         do_recfill

;*************** Box-Rahmen setzen (4) **************

        moveq       #3,d7                   ;Box Counter

        moveq       #60,d0
        moveq       #70,d1
        moveq       #62,d2
        moveq       #84,d3

CBoxLoop:
        movem.l     d0-d3,-(a7)
        bsr         do_recfill
        movem.l     (a7)+,d0-d3
        add         #59,d0
        add         #59,d2
        dbra        d7,CBoxLoop

;*************** set apen ***************************

        moveq       #3,d0
        bsr         do_Apen

;*************** SWITCH-Ground(Boxhintergrund) setzen

        bsr         CBox01
        bsr         CBox02
        bsr         CBox03
        bsr         Cbox04
        bsr         do_lfRec
        bsr         do_RiRec

;*************** Hintergrund color (Text) setzen *****

        moveq       #3,d0
        bsr         do_Bpen
        moveq       #2,d0
        jsr         setApen(a6)

;*************** Text ausgeben **********************

        bsr         do_Lftxt
        bsr         do_ritxt
        bsr         do_txt01
        bsr         do_txt02
        bsr         do_txt03
        bsr         do_txt04
        moveq       #1,d7
        bsr         ComCopy02               ;Copy-Box Invertieren

;*************** Track-Boxen ausgeben ***************
```



Listing: CLI-Copy

```

        bsr         do_SetLRTrackBox

;*************** Hauptschleife **********************

MainLoop:
        btst        #6,mousebot
        bne         IDCMP_flag

;*************** Text-Bereich loeschen **************

        moveq       #0,d0
        bsr         do_apen
        moveq       #2,d0
        moveq       #61,d1
        move        #236,d2
        moveq       #69,d3
        bsr         do_recfill

;*************** Mouse Pointer x/y Pos **************

        move.l      winhandle,a0
        move        12(a0),d1           ; ypos
        move        14(a0),d0           ; xpos
        cmp         #86,d1
        blo         ycommand
        cmp         #97,d1
        bhi         MainLoop

;*************** x-position *************************

X1pos:
        cmp         #5,d0
        blo         MainLoop
        cmp         #117,d0
        bhi         X2pos
        bsr         InvLPBox
        bra         do_findtask

X2pos:
        cmp         #125,d0
        blo         idcmp_flag
        cmp         #238,d0
        bhi         MainLoop
        bsr         InvRPBox
        bra         Mainloop

;*************** ypos-Command ***********************

YCommand:
        cmp         #72,d1
        blo         SWpos
        cmp         #3,d0
        blo         MainLoop
        cmp         #59,d0
        bhi         CX2pos
        moveq       #1,d5
        bsr         ComCopy
        moveq       #2,d0
        bsr         do_Bpen
        moveq       #112,d7
        bsr         do_Dtxt
        bra         YCommandExit

CX2pos:
        cmp         #118,d0
        bhi         CX3pos
        moveq       #2,d5
        bsr         ComCheck
        bsr         raster
        bra         YCommandExit

CX3pos:
        cmp         #178,d0
        bhi         CX4pos
        moveq       #3,d5
        bsr         ComFormat
        bsr         raster
        bra         YCommandExit

CX4pos:
        cmp         #236,d0
        bhi         MainLoop
        moveq       #4,d5
        bsr         ComName
        bsr         raster

YCommandExit:
        bsr         do_leavebot
        bra         Mainloop

;* Abfrage der MousePos fuer LaufwerksBezeichnung und
;* der Schalter Copy/Check/Format/Name

SWpos:
        cmp         #50,d1
        blo         idcmp_flag
        cmp         #58,d1
        bhi         MainLoop

SWX2Apos:
        cmp         #50,d0
        blo         MainLoop
        cmp         #80,d0
```

Listing: CLI-Copy

```
        bhi         swx2pos
        add.b       #1,DrvNr0
        cmp.b       #$34,DrvNr0
        bne         NotDrv00
        move.b      #$30,DrvNr0
NotDrv00:
        moveq       #2,d0
        bsr         do_bpen
        moveq       #0,d7
        bsr         do_Dtxt
        bsr         do_leaveBot
        clr.l       d7
        move.b      DrvNr0,d7
        eor.l       #$00000030,d7
        move.l      d7,drive
        bra         MainLoop
SWX2pos:
setnewdev:
        cmp.w       #1,ComFlag
        bne         MainLoop
        cmp         #190,d0
        bhi         MainLoop
        cmp         #160,d0
        blo         MainLoop
        add.b       #1,DrvNr
        cmp.b       #$34,DrvNr
        bne         NotDrv01
        move.b      #$30,DrvNr
NotDrv01:
        moveq       #2,d0
        bsr         do_bpen
        moveq       #112,d7
        bsr         do_Dtxt
        bsr         do_leaveBot
        clr.l       d7
        move.b      DrvNr,d7
        eor.l       #$00000030,d7
        move.l      d7,drive1
        bra         MainLoop
;*************** Schliess-Gadget Abfrage *************
IDCMp_Flag:
        move.l      execbase,a6
        move.l      winhandle,a0
        move.l      86(a0),a0
        jsr         -372(a6)
        tst         d0
        beq         MainLoop
        move.l      d0,a0
        move.l      20(a0),d0
        cmp         #$200,d0
        bne         mainloop
;*************** Programm Ende ***********************
END:
        move.l      intbase,a6
        move.l      winhandle,a0      ;close window
        jsr         closewin(a6)
;*************** close LIBS *************************
        move.l      execbase,a6
        move.l      intbase,a1
        jsr         closelib(a6)
        move.l      grabase,a1
        jsr         closelib(a6)
do_free:
        move.l      #512*11*4,d0
        move.l      bufBase,a1
        jsr         freemem(a6)
        moveq       #0,d0
        rts
;*************** SUB - Command Box (4) ***************
do_switch:
        movem.l     d0-d3,-(a7)
        bsr         do_recfill
        movem.l     (a7)+,d0-d3
        add         #110,d0
        add         #110,d2
        bsr         do_recfill
        rts
;*************** Copy/Check/Format/Name **************
;*************** Boxen zeichnen **********************
CBox01:
        moveq       #4,d0
        moveq       #57,d2
        bra         SetCbox
CBox02:
        moveq       #65,d0
        moveq       #116,d2
        bra         SetCbox
CBox03:
```

Listing: CLI-Copy

```
        moveq       #124,d0
        move        #175,d2
        bra         SetCbox
CBox04:
        move        #183,d0
        move        #234,d2
SetCBox:
        moveq       #72,d1
        moveq       #82,d3
        bra         do_recfill
;*************** Begin Process/Abort Process *********
;*************** Boxen zeichnen **********************
do_lfRec:
        move        #3,d0
        move        #86,d1
        move        #116,d2
        move        #97,d3
        bra         do_recfill
do_RiRec:
        move        #124,d0
        move        #86,d1
        move        #235,d2
        move        #97,d3
do_recfill:
        move.l      grabase,a6
        move.l      rastport,a1
        jsr         RecFill(a6)
        rts
;*************** Linke/Rechte Tracks *****************
;*************** Boxen zeichnen **********************
do_SetLRTrackBox:
        moveq       #3,d0
        bsr         do_Apen
;*************** TRAK 0/1 Felder setzen **************
        moveq       #1,d6             ; counter 2-Felder
        moveq       #30,d4            ; xpos01
boxloop02:
        move        d4,d0
        move        d4,d2
        addq        #4,d2
        moveq       #16,d1
        moveq       #18,d3
        moveq       #9,d5             ;counter x-achse 10
        moveq       #7,d7             ;counter y-achse 8
        move.l      rastport,a1
BoxLoop:
        movem.l     a1/d0-d3,-(a7)
        jsr         RecFill(a6)
        movem.l     (a7)+,a1/d0-d3
        add         #7,d0
        add         #7,d2
        dbra        d5,boxloop
        move        d4,d0
        move        d4,d2
        addq        #4,d2
        moveq       #9,d5
        addq        #4,d1
        addq        #4,d3
        dbra        d7,BoxLoop
        move        #140,d4
        dbra        d6,boxloop02
        moveq       #1,d0
do_Apen:
        move.l      grabase,a6
        move.l      rastport,a1
        jsr         setApen(a6)
        rts
do_Bpen:
        move.l      grabase,a6
        move.l      rastport,a1
        jsr         setBpen(a6)
        rts
;*************** Line ********************************
do_line:
        move.l      rastport,a1
        clr         36(a1)
        move        #237,d0
        jsr         -246(a6)
        rts
;*************** Begin/Abort - Text setzen ***********
do_lftxt:
        move.l      rastport,a1
        move        #11,36(a1)
        lea         txt1,a0
        bra         do_set
```



Listing: CLI-Copy

```
do_ritxt:
        move.l      rastport,a1
        move        #128,36(a1)
        lea         txt2,a0

do_set:
        move        #94,38(a1)
        moveq       #13,d0
        jsr         text(a6)
        rts

;*************** Copy/Check/Format/Name **************
;*************** Text setzen *************************

do_txt01:
        move.l      rastport,a1
        move        #8,36(a1)
        lea         txt01,a0
        bra         do_settxt

do_txt02:
        move.l      rastport,a1
        move        #69,36(a1)
        lea         txt02,a0
        bra         do_settxt

do_txt03:
        move.l      rastport,a1
        move        #125,36(a1)
        lea         txt03,a0
        bra         do_settxt

do_txt04:
        move.l      rastport,a1
        move        #193,36(a1)
        lea         txt04,a0
        moveq       #5,d0
        bra         do_settxt02

do_settxt:
        moveq       #6,d0

do_settxt02:
        move        #80,38(a1)
        jsr         text(a6)
        rts

;*************** DF0:/DF1: - Text setzen *************

do_Dtxt:
        moveq       #1,d0
        bsr         do_apen
        lea         Dtxt01,a0
        move.l      rastport,a1
        move        #50,36(a1)
        tst         d7
        beq         do_setDtxt
        add         d7,36(a1)
        lea         Dtxt02,a0

do_setDtxt:
        moveq       #4,d0

do_setErrTxt:
        move        #56,38(a1)
        jsr         text(a6)
        rts

;************** Process Box Invertieren **************

InvLPBox:
        moveq       #3,d0
        moveq       #114,d4
        bra         InvRPBox02

InvRPBox:
        moveq       #124,d0
        moveq       #112,d4

InvRPBox02:
        moveq       #86,d1
        moveq       #12,d5
        moveq       #0,d7                       ;flag process box

do_Cblit:
        move        d0,d2
        move        d1,d3
        move        #$30,d6
        move.l      grabase,a6
        move.l      rastport,a0
        move.b      #$f1,24(a0)
        move.l      a0,a1
        movem.l     d0-d7/a0-a1,-(a7)
        jsr         -552(a6)
        movem.l     (a7)+,d0-d7/a0-a1
        tst         d7     ;flag 0=Processbox 1=Commandbox
        bne         CblitExit
        bsr         do_leavebot
        jsr         -552(a6)

CblitExit:
        move        #0,comflag02
        move.l      rastport,a0
        move.b      #-1,24(a0)
```



Listing: CLI-Copy

```
        rts

;*************** Copy/Check/Format/Name Invertieren **

BlitBox01:
        moveq       #4,d0
        moveq       #54,d4
        bra         BlitBox1

BlitBox02:
        moveq       #65,d0
        bra         BlitBox

BlitBox03:
        moveq       #124,d0
        bra         BlitBox

BlitBox04:
        move        #183,d0

BlitBox:
        moveq       #52,d4

BlitBox1:
        cmp         #1,Comflag02
        beq         CblitExit
                    ;wenn Commando gleich dann exit
        moveq       #72,d1
        moveq       #11,d5
        moveq       #1,d7               ;flag commandbox
        bra         do_Cblit

;*************** COPY/CHECK/FORMAT/NAME Ausfuehren ***

do_findtask:
        bsr         do_SetLRtrackBox
        lea         ErrTxt01,a5
        move.l      $4,a6
        sub.l       a1,a1
        jsr         -294(a6)
        move.l      d0,diskrep+16

do_addport:
        lea         diskrep,a1
        jsr         -354(a6)

do_openDev:
        lea         diskio,a1
        move.l      #diskrep,14(a1)
        move.l      drive,d0
        clr.l       d1
        lea         DevName,a0
        jsr         -444(a6)
        tst         d0
        bne         error

do_findtask1:
        sub.l       a1,a1
        jsr         -294(a6)
        move.l      d0,diskrep1+16

do_addport1:
        lea         diskrep1,a1
        jsr         -354(a6)

do_openDev1:
        lea         diskio1,a1
        move.l      #diskrep1,14(a1)
        move.l      drive,d0
        cmp.w       #1,comflag
        bne         NoCopy
        move.l      drive1,d0

NoCopy:
        clr.l       d1
        lea         DevName,a0
        jsr         -444(a6)
        tst         d0
        bne         error02

        lea         diskio1,a1      ;Disk in drive test
        move.l      $18(a1),a1
        btst        #1,$40(A1)
        beq         WP
        lea         ErrTxt03,a5
        bsr         Do_closeDev
        bra         Error03

;*************** WriteProtect Test *******************

WP:
        clr         ErrFlag
        btst        #4,$40(A1)
        beq         NoWP
        move.b      #-1,ErrFlag

NoWP:
        moveq       #79,d6              ; 80 Tracks
        moveq       #0,d5
        clr.l       d7                  ; Start Track
        move.w      #0,a3               ; Read Error flag

;**************** COMMAND-Flag pruefen ***************

CommandCheck:
```

Listing: CLI-COPY

```
        cmp         #1,Comflag
        beq         do_Copy
        cmp         #2,ComFlag
        beq         do_CheckDisk
        cmp         #3,Comflag
        beq         do_format
        bra         do_name

;*************** COPY DISK ***************************

do_copy:
        lea         ErrTxt05,a5
        move.l      drive,d0
                   ;Test ob LaufwerksBezeichnungen ungleich
        cmp.l       drive1,d0
        bne         do_copy01

do_Error03:
        bsr         do_closeDev
        bra         error03

do_copy01:
        lea         ErrTxt04,a5    ;Test ob Disk im Laufwerk
        lea         diskio,a1
        move.l      $18(a1),a1
        btst        #1,$40(a1)
        bne         do_error03

do_copy02:
        lea         ErrTxt02,a5
                    ; Test ob WriteProtection offen
        cmp.l       #0,drive
        bne         CDr
        cmp.b       #0,errflag
        beq         do_read
        bra         do_Error03

CDr:
        cmp.b       #0,errflag
        bne         do_Error03

do_read:
        cmp         #3,ComFlag
        beq         do_write                ;Format-Flag
        bsr         ReadTrack               ;Track einlesen
        beq         NoErr                   ;auf Error testen
        move        #1,a3                   ;Error-Flag

NoErr:
        eor         #1,d5           ;Buffer-Flag Invertieren
        bsr         LtrackBox       ;gelesen Track anzeigen

        btst        #6,mousebot             ;Abort Test
        bne         do_write

        bsr         Abort_Test
        cmp.l       #"EXIT",d7
        beq         CommandExit

;*************** Write #1 ****************************

do_write:
        tst.l       d7
        beq         do_write02
        bsr         do_checkio

do_write02:
        lea         DiskIO1,a1
        move        #11,28(a1)
        move.l      bufbase,40(a1)
        tst         d5                      ;Buffer-Flag testen
        bne         buf1                    ;Buffer wechseln
        add.l       #$2c00,40(a1)

buf1:
        move.l      #512*11*2,36(a1)
        move.l      d7,44(a1)
        move.l      execbase,a6             ;Track schreiben
        jsr         -462(a6)
        bne         NoErr02                 ;Error Test
        move        #1,a3

NoErr02:
        cmp         #3,ComFlag
        bne         NoFormat02  ;geschrieben Track anzeigen
        bsr         LtrackBox
        bra         MouseWait

noFormat02:
        bsr         RtrackBox

;*************** Abort Process Test *****************

mousewait:
        btst        #6,mousebot  ;Abort Process aktiviert?
        bne         MainCopyLoop
        bsr         Abort_test
        cmp.l       #"EXIT",d7   ;Ja dann EXIT-Flag setzen
        beq         CommandExit

MainCopyLoop:
        add.l       #512*11*2,d7
        dbra        d6,do_read
        bra         CommandExit

```

Listing: CLI-Copy

```
;*************** CHECK DISK **************************

do_CheckDisk:
        bsr         readtrack
        beq         NoErrCD
        move        #1,a3

NoErrCD:
        add.l       #512*11*2,d7
        bsr         LtrackBox

        btst        #6,mousebot
        bne         CheckDiskLoop
        bsr         Abort_test
        cmp.l       #"EXIT",d7
        beq         CommandExit02

CheckDiskLoop:
        dbra        d6,do_checkDisk
        bra         CommandExit02

;*************** FORMAT ******************************

do_format:
        move.l      #$1600-1,d0
        move.l      bufbase,a5

ClrLoop:
        move.l      #$0,(a5)+
        dbra        d0,clrloop
        moveq       #24,d0
        move.l      Bufbase,a5
        lea         BootBlock,a4

BBloop:
        move.w      (a4)+,(a5)+
        dbra        d0,BBloop
        lea         ErrTxt02,a5

        cmp.l       #0,drive
        bne         C2
        cmp.b       #0,errflag
        beq         DrOk
        bra         DrErr

c2:
        cmp.b       #0,errflag
        beq         DrOk

DrErr:
        bsr         do_closeDev
        bra         error03

DrOk:
        moveq       #1,d5           ;bufzeiger auf buffer0
        bsr         do_write

;*************** Check auf Abort ! *******************

        moveq       #0,d5           ;bufzeiger auf buffer1
        cmp.l       #"EXIT",d7
        beq         ExitDev

        move.l      bufbase,a5
        add.l       #$2c00,a5
        move.l      a5,a4
        add.l       #$200,a4

        moveq       #55,d0           ;counter Bitmap

clrBitMap:
        move.l      #-1,(a4)+
        dbra        d0,ClrBitMap

;*************** Root-Dir und Dir-Bitmap *************

        move.w      #2,2(a5)
        move.b      #$48,15(a5)
        move.l      #$b0bbb2f0,20(a5)
        move.w      #1,$13a(a5)
        move.w      #$0371,$13e(a5)
        move.b      #4,$1b0(a5)
        move.b      #"D",$1b1(a5)
        move.b      #"I",$1b2(a5)
        move.b      #"S",$1b3(a5)
        move.b      #"K",$1b4(a5)
        move.b      #1,$1ff(a5)
        move.l      #$c000c037,$200(a5)
        move.b      #$3f,$272(a5)
        move.b      #$3f,$2dc(a5)
        move.l      #$2c00*40,d7            ;track 40 setzen
        moveq       #0,d6
        bra         do_write02

;*************** NAME ********************************

do_Name:
        move.l      #$2c00*40,d7
        bsr         readtrack
        move        d0,d6                   ;Error Retten

        bsr         do_setErrCol

        move.l      rastport,a1
        move        #8,36(a1)
```

Listing: CLI-Copy

```
        move        #68,38(a1)
        move.l      bufbase,a0
        moveq       #0,d0
        move.b      $1b0(a0),d0
        add.w       #$1b1,a0
        cmp.w       #26,d0
        blo         NameLenghOK
        moveq       #26,d0

NameLenghOK:
        tst         d6
        beq         NoNameErr
        lea         NameErr,a0
        moveq       #10,d0

NoNameErr:
        move.l      grabase,a6
        jsr         text(a6)
        bra         CommandExit02

do_setErrCol:
        moveq       #0,d0
        bra         do_bpen

;*************** Copy/Check/Format/Name AUSGANG ******

CommandExit:
        bsr         do_checkIO

;*Track Parameter wieder in Grundeinstellung versetzen

CommandExit02:
        move.l      #$001e0010,xpos1  ;(fuer GrafikAusgabe)
        move.l      #$00220012,ypos1

;*************** Motor ausstellen ********************

do_Moff:
        move.l      execbase,a6
        lea         diskio,a1
        move        #9,28(a1)
        move.l      #0,36(a1)
        jsr         -456(a6)
        lea         diskio1,a1
        move        #9,28(a1)
        move.l      #0,36(a1)
        jsr         -456(a6)

        cmp.w       #3,Comflag
        bne         ExitDev
        cmp         #1,d5
        beq         exit

;*************** TrackDevice wieder schliessen *******

ExitDev:
        bsr         do_remPort
        bsr         do_closeDev
        cmp         #4,ComFlag
        beq         MainLoop
                    ;Endtxt nicht ausgeben, da NAME
        lea         ENDtxt,a5
        cmp.l       #"EXIT",d7
        bne         PFin
        lea         EndTxt02,a5

PFin:
        bsr         Settxt
        bra         MainLoop

;*************** Do_CloseDev *************************

do_closeDev:
        lea         diskio1,a1
        jsr         -450(a6)

do_closeDev02:
        lea         diskio,a1
        jsr         -450(a6)
        rts

;*************** Do_RemPort **************************

do_remPort:
        lea         diskrep1,a1
        jsr         -360(a6)

do_RemPort02:
        lea         diskrep,a1
        jsr         -360(a6)

Exit:
        rts

;*************** Test ob BeginIO fertig **************

do_checkIO:
        move.l      $4,a6
        lea         diskio1,a1
        jsr         -468(a6)
        beq         do_CheckIO
        rts

;*************** Read track **************************

```

Listing: CLI-Copy

```
Readtrack:
        lea         DiskIO,a1

readtrack01:                            ;EinSprung Check/Name
        move.l      $18(a1),a0
        move.b      #1,$34(a0)          ;retry count =1
        move.l      $4,a6
        move.w      #2,28(a1)
        move.l      bufbase,40(a1)
        tst         d5
        beq         buf0
        add.l       #$2c00,40(a1)

buf0:
        move.l      #512*11*2,36(a1)    ;Track Anzahl
        move.l      d7,44(a1)           ;Start Track
        jsr         -456(a6)
        move.l      $18(a1),a0
        move.b      #10,$34(a0)
        tst.l       d0
        rts

;*************** ABORT-TEST **************************

Abort_Test:
        move.l      winhandle,a0
        move        12(a0),d1           ; ypos
        move        14(a0),d0           ; xpos

        cmp         #125,d0
        blo         exit
        cmp         #238,d0
        bhi         exit
        move.w      d5,-(a7)
        bsr         InvRPBox
        move.w      (a7)+,d5            ;bufflag retten
        move.l      #"EXIT",d7
        rts

;*************** Track Boxen  Setzen!!! **************

LTrackBox:
        bsr         AddTrackPos
        bsr         do_recfill
        cmp         #1,ComFlag
        beq         exit
        bra         addpos

RTrackbox:
        bsr         AddTrackPos
        add         #110,d0
        add         #110,d2
        bsr         do_recfill

addpos:
        add         #7,xpos1
        add         #7,ypos1
        cmp         #$64,xpos1
        bne         Exit

        move        #30,xpos1
        move        #34,ypos1
        add         #4,xpos2
        add         #4,ypos2
        rts

;*************** AddTrack X/Y Postion ****************

AddTrackPos:
        move.w      xpos1,d0
        move.w      xpos2,d1
        move.w      ypos1,d2
        move.w      ypos2,d3
        cmp         #0,a3
        beq         Exit
        addq        #1,d0
        addq        #1,d1
        subq        #1,d2
        subq        #1,d3
        move        #0,a3
        rts

;*************** ERROR Ausgang ***********************

Error:
        bsr         do_RemPort02
        bra         SetErrTxt

Error02:
        bsr         do_CloseDev02

error03:
        bsr         do_remPort

SetErrTxt:
        bsr         setTxt
        bra         MainLoop

SetTxt:
        bsr         do_setErrCol
        move.l      a5,a0
        move.l      rastport,a1
        move        #8,36(a1)
        move        #68,38(a1)
        move        #25,d0
```



Listing: CLI-COPY

```
        jsr         text(a6)
        move        #3,d0
        bra         do_bpen
;*************** Leave MouseBot ***********************
do_LeaveBot:
        btst        #6,mousebot
        beq         do_LeaveBot
        rts
;***** Fill Process Box - Command-Box neu Setzen  ****
ComCopy:
        bsr         TestCom
ComCopy02:
        bra         BlitBox01
ComCheck:
        bsr         TestCom
        bra         BlitBox02
ComFormat:
        bsr         TestCom
        bra         BlitBox03
ComName:
        bsr         testcom
        bra         Blitbox04
;*************** Test, ob neues Commando gesetzt *****
TestCom:
        move        ComFlag,d7          ;altes Flag retten
        move        d5,ComFlag
        cmp         d5,d7
        bne         NewCom
        move        #1,Comflag02
        bra         TestComExit
NewCom:
        cmp         #1,d7
        bne         tc2
        bsr         BlitBox01
        rts
tc2:
        cmp         #2,d7
        bne         tc3
        bsr         BlitBox02
        rts
tc3:
        cmp         #3,d7
        bne         tc4
        bsr         BlitBox03
        rts
TC4:
        bsr         BlitBox04
TestComExit:
        rts
raster:
        moveq       #2,d0
        bsr         do_apen
        moveq       #51,d1
rastloop:
        move.l      rastport,a1
        move        #162,36(a1)
        move        d1,38(a1)
        move        #184,d0
        move.l      d1,-(a7)
        jsr         -246(a6)
        move.l      (a7)+,d1
        addq        #2,d1
        cmp         #57,d1
        bne         rastloop
        rts
;*************** PARAMETER ***************************
drive:
        dc.l        0
drivel:
        dc.l        1
bufbase:
        dc.l        0
intbase:
        dc.l        0
grabase:
        dc.l        0
winHandle:
        dc.l        0
rastport:
        dc.l        0
;*************** WINDOW DEF **************************
window_def:
        dc.w        200                 ;x-pos
```

Listing: CLI-Copy

```
        dc.w        20                  ;y-pos
        dc.w        240                 ;bright
        dc.w        100                 ;high
        dc.b        2                   ;text-pen
        dc.b        1                   ;ground-pen
        dc.l        $200                ;IDCMP-flag
        dc.l        $100a               ;active-flag
        dc.l        0                   ;first-gad
        dc.l        0                   ;checkMark
        dc.l        Wtxt
SCRhandle:
        dc.l        0
        dc.l        0                   ;bitmap
        dc.w        150                 ;minbright
        dc.w        50                  ;minHigh
        dc.w        640                 ;maxBright
        dc.w        200                 ;maxHigh
        dc.w        15                  ;screen-type
        even
;*************** X/Y Positon von 1.ter Track Box *****
xpos1:
        dc.w        30
xpos2:
        dc.w        16
ypos1:
        dc.w        34
ypos2:
        dc.w        18
        even
wtxt:
        dc.b " CLI-COPY V1.09 (c) DCR ",0
ErrTxt01:
        dc.b "Device Not Present !       "
ErrTxt02:
        dc.b "Remove Write-Protection !"
ErrTxt03:
        dc.b "No Destination Disk !      "
ErrTxt04:
        dc.b "No Source Disk !           "
ErrTxt05:
        dc.b "Wrong Destination Drive !"
EndTxt:
        dc.b "Process Finished !         "
EndTxt02:
        dc.b "User Break !               "
NameErr:
        dc.b "READ ERROR"
DevName:
        dc.b "trackdisk.device",0
intname:
        dc.b "intuition.library",0
graname:
        dc.b "graphics.library",0
        even
txt1:
        dc.b "START PROCESS"
txt2:
        dc.b "ABORT PROCESS"
txt01:
        dc.b " COPY "
txt02:
        dc.b "CHECK "
txt03:
        dc.b "FORMAT"
txt04:
        dc.b "NAME  "
Dtxt01:
        dc.b "DF"
drvnr0:
        dc.b "0:"
Dtxt02:
        dc.b "DF"
drvnr:
        dc.b "1:"
SDtxt01:
        dc.b "S"
SDtxt02:
        dc.b "D"
        even
comflag:
        dc.w        1
comflag02:
        dc.w        0
Errflag:
        dc.w        0
diskio:
        blk.l       20,0
diskrep:
        blk.l       8,0
diskio1:
        blk.l       20,0
diskrep1:
        blk.l       8,0
BootBlock:
dc.w $444f,$5300,$c020,$0f19,$0000,$0370,$43fa,$0018
dc.w $4eae,$ffa0,$4a80,$670a,$2040,$2068,$0016,$7000
dc.w $4e75,$70ff,$60fa,$646f,$732e,$6c69,$6272,$6172
dc.w $7900
;*************** Ende von CLI-Copy ******************
```

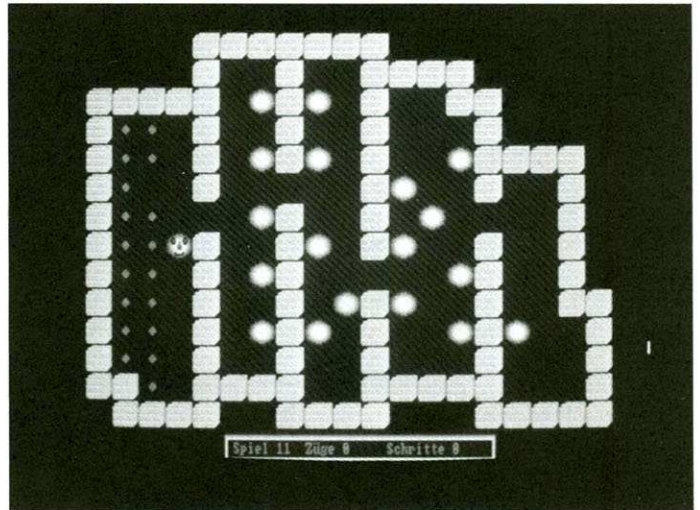
Listing: CLI-Copy

Rolf Wagner

Das Spiel der Arbeit

Hard-Work

Hier wird keine ruhige Kugel geschoben, ganz im Gegenteil. Joystick-Action ist angesagt.



Schaffen wir doch etwas Ordnung

"Spielen bis der Joystick glüht" ist die Devise. Hard-Work ist ein Spiel, das die grauen Zellen ins Schwitzen bringt. Die Spielidee ist relativ einfach: Es müssen lediglich Kugeln auf vorbestimmte Ziele geschoben werden, das war's dann auch schon. Aber ganz so einfach, wie es klingt, ist es nun auch nicht, denn wie bereits erwähnt, muß man sein Hirn doch etwas anstrengen, um Fehler zu vermeiden. Fallen für solche Fehler halten die verschiedenen Levels jederzeit parat.

Hard-Work ist eine Umsetzung des bekannten Spielhits Sokoban, das auf den PCs beachtliche Erfolge erzielen konnte. Mit dem eingebauten Spiele-Editor von Hard-Work lassen sich einige zusätzliche Levels erstellen.

Der Kreativität sind damit so gut wie keine Grenzen gesetzt. Das sorgt natürlich auch für langanhaltenden Spielspaß, wenn das Wetter draußen mal nicht den eigenen Vorstellungen entspricht. Mit dem Spiele-Editor können auch alte Spielvariationen verändert werden. Er ist voll mausgesteuert. Bei den verschiedenen Optionen des Editors sind Setzen und Löschen der Figuren, Testen, Abspeichern und Beenden möglich. Nach dem Testen verlassen Sie den Editor mit der Taste [Esc].

Haben Sie bei den unterschiedlichen Levels den richtigen Weg nicht gefunden, hilft Ihnen die Escape-Taste aus dieser Bredouille heraus, und Sie gelangen wieder ins Hauptmenü zurück. Sollten Sie eine Kugel falsch geschoben haben, drücken Sie einfach die Taste [Backspace] für eine Zugzurücknahme. Eine Zug- und Spielanzeige wird am unteren Bildschirmrand ausgegeben.

Mit Hilfe des Programmgenerators können die ersten drei Spiele generiert werden. Das Programm »GEN« muß dabei vor dem ersten Start von Hard-Work aufgerufen werden. Jedoch sollte man mit diesem Programm die Vorsicht walten lassen, denn bei jedem erneuten Aufruf wird die gegebenenfalls bestehende Datei überschrieben.

Mit dem Programm »DRUCK« hat man die Möglichkeit, sich die Spielfelder wahlweise am Bildschirm oder auf dem Drucker ausgeben zu lassen. Wie jedes AmigaBASIC-Listing kann auch das Spiel Hard-Work mit Hilfe von [CTRL] + [C] beendet werden. Auf unserer Databox finden Sie neben der lauffähigen Version von Hard-Work eine Spieledatei, die bereits 34 fertige Levels enthält. Diese Levels haben natürlich unterschiedliche Schwierigkeitsgrade und sorgen für anhaltenden Spielspaß. Viel Freude beim "Fraggeln" mit Hard-Work.

(vb)

Listings

```
REM ***************************************************
REM *                  HARD-WORK                      *
REM *              Sokoban-Umsetzung                  *
REM *               by Rolf Wagner                    *
REM *             (c) 1990 DMV-Verlag                 *
REM *            Sprache: AMIGA BASIC                 *
REM ***************************************************

 DEFINT a-w
 CLEAR,30000
 ma=6:zi=12:ku=7:ca=14:m=0:ziele=0:spiele=0:zug=0
 schritt=0
 DIM spiele$(35),spiele(35)
 DIM leer%(130),ziel%(130),mauer%(130),kugel%(130)
,face%(130)
 SCREEN 1,639,271,4,2:WINDOW CLOSE 1:WINDOW 2,,,0,1
 PALETTE 0,.1,.1,.1:PALETTE 1,.1,.1,.1
 PALETTE 2,.1,.1,.1:PALETTE 3,.2,.2,.2
 PALETTE 4,.3,.3,.3:PALETTE 5,.4,.4,.4
 PALETTE 6,.5,.5,.5:PALETTE 7,.6,.6,.6
 PALETTE 8,.99,.5,0:PALETTE 9,.8,.8,.8
 PALETTE 10,.9,.9,.9:PALETTE 11,1,1,1
 PALETTE 12,1,0,0:PALETTE 13,1,.2,0
 PALETTE 14,.1,.1,.7:PALETTE 15,.4,.1,0
 x1=150:x2=460:y1=42:y2=60
 LINE (x1,y1)-(x2,y2),3,b
 GOSUB fenster
 COLOR 7:LOCATE 7,22
 PRINT "H A R D - W O R K"
 COLOR 12:LOCATE 7,41
 PRINT "Spiele laden"
 ON ERROR GOTO violation
 GOSUB laden
 CLS
 POKEW 14676118&,384
leerfigur:
 GET(0,0)-(30,15),leer%
mauerfigur:
 LINE (0,0)-(30,14),8,bf
 FOR i=3 TO 15 STEP 4
  LINE (0,0+i)-(30,0+i),6
 NEXT i
 LINE (24,3)-(30,3),8:LINE (24,7)-(30,7),8
```

Listing 1. HARD-WORK

```
 LINE (24,11)-(30,11),8:LINE (24,7)-(30,2),6
 LINE (24,11)-(30,6),6:LINE (15,7)-(15,11),6
 LINE (8,4)-(8,7),6:LINE (8,12)-(8,14),6
 LINE (23,4)-(23,7),6:LINE (23,12)-(23,14),6
 LINE (14,0)-(6,3),6:LINE (22,0)-(14,3),6
 LINE (23,3)-(29,0),6:LINE (23,3)-(23,13),6
 LINE (6,0)-(0,2),1:LINE (5,0)-(0,1),1
 LINE (4,0)-(0,0),1:LINE (24,15)-(30,13),1
 LINE (25,15)-(30,14),1:LINE (26,15)-(30,15),1
 LINE (27,15)-(30,15),1:LINE (30,0)-(30,15),1
 LINE (0,15)-(23,15),1:LINE (24,15)-(30,12),1
 LINE (1,3)-(23,3),6:LINE (26,1)-(26,13),6
 LINE (27,1)-(27,12),6:LINE (3,1)-(25,1),6
 PSET (0,14),0:PSET (9,2),8:PSET (18,2),8
 GET(0,0)-(30,15),mauer%
kugelfigur:
 LINE (0,0)-(32,15),0,bf
 x=15:y=7:z=.44
 CIRCLE (x,y),15,3,,,z:CIRCLE (x,y),14,3,,,z
 CIRCLE (x,y),13,4,,,z:CIRCLE (x,y),12,4,,,z
 CIRCLE (x,y),11,5,,,z:CIRCLE (x,y),10,5,,,z
 CIRCLE (x,y),9,6,,,z:CIRCLE (x,y),8,6,,,z
 CIRCLE (x,y),7,7,,,z
 PAINT (x,y),7
 GET(0,0)-(30,15),kugel%
Sprite:
 COLOR 0
 LINE(7,3)-(10,6),,b:LINE(20,3)-(23,6),,b
 LINE(6,4)-(6,5):LINE(11,4)-(11,5)
 LINE(19,4)-(19,5):LINE(24,4)-(24,5)
 COLOR 12
 LINE(8,4)-(9,5),,b:LINE(21,4)-(22,5),,b
 COLOR 14
 LINE(14,5)-(16,9),,bf
 LINE(13,7)-(13,9):LINE(17,7)-(17,9)
 PSET(15,4),14:PSET(12,8),14:PSET(18,8),14
 COLOR 15
 LINE(10,11)-(12,11):LINE(18,11)-(20,11)
 LINE(12,12)-(18,12)
 GET(0,0)-(30,15),face%
Zielfigur:
 LINE (0,0)-(32,15),0,bf
 CIRCLE(15,7),4,12,,,.48
 PAINT (15,7),12
 GET(0,0)-(30,15),ziel%
 CLS
```

Listing 1. HARD-WORK

```
 POKEW 14676118&,33664&
start:
 CLS
 m=0:testphase=0:ziele=0:zug=0:schritt=0:te=0
 spiele=maxa:wahl$=""
 x1=150:x2=460:y1=42:y2=60
 LINE (x1,y1)-(x2,y2),3,b
 GOSUB fenster
 COLOR 4:LOCATE 7,30
 PRINT "H A R D - W O R K"
 COLOR 12
 y1=66:y2=84
 LINE (x1,y1)-(x2,y2),3,b
 GOSUB fenster
 LOCATE 10,25
 PRINT "Es sind";spiele;"Spiele vorhanden"
 y1=90:y2=108
 LINE (x1,y1)-(x2,y2),3,b
 GOSUB fenster
 LOCATE 13,30
 INPUT "S)piel oder E)ditor";wahl$
 IF wahl$="s" THEN
  y1=114:y2=132
  LINE (x1,y1)-(x2,y2),3,b
  GOSUB fenster
  LOCATE 16,24
  INPUT "Welches Spiel moechten Sie";spiel
   IF spiel>spiele THEN spiel=1
  spiele=spiel
 END IF
 IF wahl$="e" THEN
  y1=114:y2=132
  LINE (x1,y1)-(x2,y2),3,b
  GOSUB fenster
  LOCATE 16,24
  INPUT "Welches Spiel editieren";spiel
   IF spiel>spiele THEN spiel=spiele+1
  spiele=spiel
 END IF
 IF wahl$<>"s" AND wahl$<>"e" THEN GOTO start
 IF spiele<1  THEN GOTO start
 IF wahl$="s" THEN GOSUB zeichnen:GOTO test
 IF spiele>maxa THEN
  CLS
  IF LEN(spiele$(spiele))>1 THEN GOSUB zeichnen
  GOTO editor
 END IF
 GOSUB zeichnen
 GOTO editor
fenster:
 LINE (x1+1,y1+1)-(x2-1,y2-1),5,b
 LINE (x1+2,y1+2)-(x2-2,y2-2),6,b
 LINE (x1+3,y1+3)-(x2-3,y2-3),7,b
 RETURN
editor:
 x1=1:y1=242:x2=630:y2=260
 LINE (0,y1)-(x2,y2),0,bf:LINE (0,y1)-(x2,y2),3,b
 GOSUB fenster
 COLOR 12:LOCATE 32,2
 PRINT "EDITOR:";
 COLOR 6:LOCATE 32,10
 PRINT "Mauer Kugel Face Ziel Loeschen Test
Speichern Ende";
 COLOR 12:LOCATE 32,60
 PRINT "Spiel:";spiele;"von";maxa
 LOCATE 32,79
eingabe:
 dummy=MOUSE(0)
 IF MOUSE(0)=0 THEN eingabe
 x=MOUSE(1):y=MOUSE(2)
 COLOR 13
 IF y>247 AND y<255 THEN
  IF x>75 AND x<114 THEN
   LOCATE 32,10:PRINT "Mauer":GOSUB warte1:GOTO mauer
  END IF
  IF x>123 AND x<160 THEN
   LOCATE 32,16:PRINT "Kugel":GOSUB warte1:GOTO kugel
  END IF
  IF x>171 AND x<202 THEN
   LOCATE 32,22:PRINT "Face":GOSUB warte1:GOTO fraze
  END IF
  IF x>211 AND x<240 THEN
   LOCATE 32,27:PRINT "Ziel":GOSUB warte1:GOTO ziel
  END IF
  IF x>251 AND x<306 THEN
   LOCATE 32,32:PRINT "Loeschen":GOSUB warte1
   GOTO loeschen
  END IF
  IF x>317 AND x<345 THEN
   LOCATE 32,40:PRINT "Test":GOSUB warte1
   GOSUB speichern:testphase=1:GOTO test
  END IF
  IF x>355 AND x<426 THEN
   LOCATE 32,45:PRINT "Speichern":GOSUB warte1
   GOTO abspeichern
  END IF
  IF x>434 AND x<466 THEN
   GOSUB warte1
    IF spiele>maxa THEN CLS:GOTO start
   GOTO start
  END IF
 END IF
 GOTO eingabe
mauer:
 dummy=MOUSE(0)
 IF MOUSE(0)=0 THEN mauer
 x=MOUSE(3):y=MOUSE(4)
 IF y>247 AND y<255 THEN
  IF x>75 AND x<114 THEN
   COLOR 6:LOCATE 32,10:PRINT "Mauer":GOSUB warte1
   GOTO eingabe
  END IF
 END IF
 IF x>619 OR y>239 THEN GOTO mauer
 xx=(x+31)/31:xx=INT(xx)-1:yy=(y+16)/16:yy=INT(yy)-1
 PUT(xx*31,yy*16),mauer%,PSET
 IF posx=xx*31 AND posy=yy*16 THEN setflag=0
 GOTO mauer
kugel:
 dummy=MOUSE(0)
 IF MOUSE(0)=0 THEN kugel
 x=MOUSE(3):y=MOUSE(4)
 IF y>247 AND y<255 THEN
  IF x>123 AND x<160 THEN
   COLOR 6:LOCATE 32,16:PRINT "Kugel":GOSUB warte1
   GOTO eingabe
  END IF
 END IF
 IF x>619 OR y>239 THEN GOTO kugel
 xx=(x+31)/31:xx=INT(xx)-1:yy=(y+16)/16:yy=INT(yy)-1
 PUT(xx*31,yy*16),kugel%,PSET
 IF posx=xx*31 AND posy=yy*16 THEN setflag=0
 GOTO kugel
fraze:
 dummy=MOUSE(0)
 IF MOUSE(0)=0 THEN fraze
 x=MOUSE(3):y=MOUSE(4)
 IF y>247 AND y<255 THEN
  IF x>171 AND x<202 THEN
   COLOR 6:LOCATE 32,22:PRINT "Face":GOSUB warte1
   GOTO eingabe
  END IF
 END IF
 IF x>619 OR y>239 THEN GOTO fraze
 xx=(x+31)/31:xx=INT(xx)-1:yy=(y+16)/16:yy=INT(yy)-1
 IF setflag=1 THEN PUT (posx,posy),leer%,PSET
 PUT(xx*31,yy*16),face%,PSET
 posx=xx*31:posy=yy*16:setflag=1
 GOTO fraze
ziel:
 dummy=MOUSE(0)
 IF MOUSE(0)=0 THEN ziel
 x=MOUSE(3):y=MOUSE(4)
 IF y>247 AND y<255 THEN
  IF x>211 AND x<240 THEN
   COLOR 6:LOCATE 32,27:PRINT "Ziel":GOSUB warte1
   GOTO eingabe
  END IF
 END IF
 IF x>619 OR y>239 THEN GOTO ziel
 xx=(x+31)/31:xx=INT(xx)-1:yy=(y+16)/16:yy=INT(yy)-1
 PUT(xx*31,yy*16),ziel%,PSET
 IF posx=xx*31 AND posy=yy*16 THEN setflag=0
 GOTO ziel
loeschen:
 dummy=MOUSE(0)
 IF MOUSE(0)=0 THEN loeschen
 x=MOUSE(3):y=MOUSE(4)
 IF y>247 AND y<255 THEN
  IF x>251 AND x<306 THEN
   COLOR 6:LOCATE 32,32:PRINT "Loeschen":GOSUB warte1
   GOTO eingabe
  END IF
 END IF
 IF x>619 OR y>239 THEN GOTO loeschen
 xx=(x+31)/31:xx=INT(xx)-1:yy=(y+16)/16:yy=INT(yy)-1
 PUT(xx*31,yy*16),leer%,PSET
 IF posx=xx*31 AND posy=yy*16 THEN setflag=0
 GOTO loeschen
test:
 IF setflag=0 THEN
  COLOR 6:LOCATE 32,40:PRINT "Test":GOSUB warte1
  GOTO eingabe
 END IF
 IF m=ziele THEN
  SOUND 440,5,255:zug=0:schritt=0:testphase=0
   IF wahl$="s" THEN
    spiele=spiele+1
   END IF
   IF wahl$="e" THEN
    GOSUB zeichnen:GOTO editor
   END IF
   IF spiele>maxa THEN
    spiele=maxa
   END IF
  GOSUB zeichnen:GOSUB speichern
  GOTO test
 END IF
 a$=INKEY$:x1%=STICK(2):y1%=STICK(3)
  IF x1%=-1 THEN
   GOSUB altnull:GOTO links
  END IF
  IF x1%=1 THEN
   GOSUB altnull:GOTO rechts
  END IF
  IF y1%=-1 THEN
   GOSUB altnull:GOTO hoch
  END IF
  IF y1%=1 THEN
   GOSUB altnull:GOTO runter
  END IF
  IF a$=CHR$(27) THEN
   GOTO start
  END IF
  IF a$=CHR$(8) THEN
   GOTO zurueck
  END IF
 GOTO test
```



Listing 1. HARD-WORK

```
altnull:
 hoaltca=0:hoaltku=0:rualtca=0:rualtku=0
 realtku=0:realtca=0:lialtku=0:lialtca=0
 RETURN
hoch:
 z=TIMER
 WHILE TIMER-z <.05:WEND
 IF posy=0 THEN GOTO test
 ho1=POINT(posx+15,posy-7):ho2=POINT(posx+15,posy-23)
 IF ho1=ku AND POINT(posx+15,22)=ca THEN
  GOTO test
 ELSEIF ho1=ma THEN
  GOTO test
 ELSEIF ho1=ku OR ho1=11 THEN
  hoaltku=0:hoaltca=0
  IF ho2=ku OR ho2=ma OR ho2=11 THEN GOTO test
   IF ho2=zi THEN
    m=m+1
   END IF
   IF ho1=11 THEN
    m=m-1
   END IF
   SOUND 3000,.03,150
   zug=zug+1
   PUT (posx,posy-16),kugel%
   altx=posx:kalty=posy-16
   PUT (posx,posy-32),kugel%
   kneuy=posy-32:hoaltku=1
  END IF
 PUT (posx,posy),face%
 calty=posy
 PUT (posx,posy-16),face%
 cneuy=posy-16
 SOUND 180,.3,255
 IF testphase=0 THEN
  schritt=schritt+1:LOCATE 32,36
  PRINT zug;TAB(50);schritt
 END IF
 posy=posy-16:hoaltca=1
 GOTO test
runter:
 z=TIMER
 WHILE TIMER-z <.05:WEND
 IF posy=224 THEN GOTO test
 ru1=POINT(posx+15,posy+23):ru2=POINT(posx+15,posy+39)
 IF ru1=ku AND POINT(posx+15,217)=ca THEN
  GOTO test
 ELSEIF ru1=ma THEN
  GOTO test
 ELSEIF ru1=ku OR ru1=11 THEN
  rualtku=0:rualtca=0
  IF ru2=ku OR ru2=ma OR ru2=11 THEN GOTO test
   IF ru2=zi THEN
    m=m+1
   END IF
   IF ru1=11 THEN
    m=m-1
   END IF
   SOUND 3000,.03,150
   zug=zug+1
   PUT (posx,posy+16),kugel%
   altx=posx:kalty=posy+16
   PUT (posx,posy+32),kugel%
   kneuy=posy+32:rualtku=1
  END IF
 PUT (posx,posy),face%
 calty=posy
 PUT (posx,posy+16),face%
 cneuy=posy+16
 SOUND 180,.3,255
 IF testphase=0 THEN
  schritt=schritt+1:LOCATE 32,36
  PRINT zug;TAB(50);schritt
 END IF
 posy=posy+16:rualtca=1
 GOTO test
rechts:
 z=TIMER
 WHILE TIMER-z <.05:WEND
 IF posx=589 THEN GOTO test
 re1=POINT(posx+46,posy+7):re2=POINT(posx+77,posy+7)
 IF POINT (604,posy+7)=7 AND POINT (573,posy+7)=ca THEN
  GOTO test
 ELSEIF re1=ma THEN
  GOTO test
 ELSEIF re1=ku OR re1=11 THEN
  realtku=0:realtca=0
  IF re2=ku OR re2=ma OR re2=11 THEN GOTO test
   IF re2=zi THEN
    m=m+1
   END IF
   IF re1=11 THEN
    m=m-1
   END IF
   SOUND 3000,.03,150
   zug=zug+1
   PUT (posx+31,posy),kugel%
   kaltx=posx+31:alty=posy
   PUT (posx+62,posy),kugel%
   kneux=posx+62:realtku=1
  END IF
 PUT (posx,posy),face%
 caltx=posx
 PUT (posx+31,posy),face%
 cneux=posx+31
 SOUND 180,.3,255
 IF testphase=0 THEN
  schritt=schritt+1:LOCATE 32,36
  PRINT zug;TAB(50);schritt
 END IF
 posx=posx+31:realtca=1
 GOTO test
links:
 z=TIMER
 WHILE TIMER-z <.05:WEND
 IF posx=0 THEN GOTO test
 li1=POINT(posx-15,posy+7):li2=POINT(posx-46,posy+7)
 IF POINT(15,posy+7)=7 AND POINT (46,posy+7)=ca THEN
  GOTO test
 ELSEIF li1=ma THEN
  GOTO test
 ELSEIF li1=ku OR li1=11 THEN
  lialtku=0
  lialtca=0
  IF li2=ku OR li2=ma OR li2=11 THEN GOTO test
   IF li2=zi THEN
    m=m+1
   END IF
   IF li1=11 THEN
    m=m-1
   END IF
   SOUND 3000,.03,150
   zug=zug+1
   PUT (posx-31,posy),kugel%
   kaltx=posx-31:alty=posy
   PUT (posx-62,posy),kugel%
   kneux=posx-62:lialtku=1
  END IF
 PUT (posx,posy),face%
 caltx=posx
 PUT (posx-31,posy),face%
 cneux=posx-31
 SOUND 180,.3,255
 IF testphase=0 THEN
  schritt=schritt+1:LOCATE 32,36
  PRINT zug;TAB(50);schritt
 END IF
 posx=posx-31:lialtca=1
 GOTO test
zurueck:
 IF hoaltku=1 AND hoaltca=1 THEN
  PUT (altx,cneuy),face%:PUT (altx,calty),face%
  PUT (altx,kneuy),kugel%:PUT (altx,kalty),kugel%
  posy=posy+16:hoaltku=0:hoaltca=0:so=1
  IF ho2=zi THEN m=m-1
  IF ho1=11 THEN m=m+1
 END IF
 IF rualtku=1 AND rualtca=1 THEN
  PUT (altx,cneuy),face%:PUT (altx,calty),face%
  PUT (altx,kneuy),kugel%:PUT (altx,kalty),kugel%
  posy=posy-16:rualtku=0:rualtca=0:so=1
  IF ru2=zi THEN m=m-1
  IF ru1=11 THEN m=m+1
 END IF
 IF realtku=1 AND realtca=1 THEN
  PUT (cneux,alty),face%:PUT (caltx,alty),face%
  PUT (kneux,alty),kugel%:PUT (kaltx,alty),kugel%
  posx=posx-31:realtku=0:realtca=0:so=1
  IF re2=zi THEN m=m-1
  IF re1=11 THEN m=m+1
 END IF
 IF lialtku=1 AND lialtca=1 THEN
  PUT (cneux,alty),face%:PUT (caltx,alty),face%
  PUT (kneux,alty),kugel%:PUT (kaltx,alty),kugel%
  posx=posx+31:lialtku=0:lialtca=0:so=1
  IF li2=zi THEN m=m-1
  IF li1=11 THEN m=m+1
 END IF
 FOR i=100 TO 1000 STEP 100:SOUND i,.55:NEXT i
 IF so=1 THEN
  FOR i=1000 TO 100 STEP -100:SOUND i,.55:NEXT i
  so=0
 END IF
 GOTO test
laden:
 OPEN "Dat" FOR INPUT AS#1
 INPUT #1,maxa
 FOR i=1 TO maxa
  LOCATE 7,53:PRINT i:LINE INPUT #1,spiele$(i)
 NEXT i
 CLOSE 1:COLOR 12
 RETURN
speichern:
 POKEW 14676118&,384
 k=1:ziele=0:merkca=0
 IF LEN(spiele$(spiele))>1 THEN spiele$(spiele)=""
 FOR i=0 TO 224 STEP 16
  FOR j=0 TO 589 STEP 31
   p=POINT(j+15,i+7)
    IF p=12 THEN
     p=2:ziele=ziele+1
    END IF
    IF p=14 THEN
     p=4:posx=j:posy=i:merkca=1
    END IF
   spiele$(spiele)=spiele$(spiele)+CHR$(p+48)
  NEXT j
 NEXT i
 setflag=1:m=0
 IF ziele=0 OR merkca=0 THEN GOTO editor
 IF wahl$="e" THEN
  COLOR 6:LOCATE 32,40:PRINT "Test":te=1
 END IF
 POKEW 14676118&,33664&
 RETURN
```

Listing 1. HARD-WORK

DATABOX

```
abspeichern:
 GOSUB speichern
 IF spiele>maxa THEN maxa=maxa+1
 OPEN "Dat" FOR OUTPUT AS#1
 PRINT #1,maxa
 FOR i=1 TO maxa:PRINT #1,spiele$(i):NEXT i
 CLOSE 1:COLOR 6
 LOCATE 32,45:PRINT "Speichern"
 GOTO editor
zeichnen:
 POKEW 14676118&,384
 ziele=0:k=1
 CLS
 FOR i=0 TO 224 STEP 16
  FOR j=0 TO 589 STEP 31
  wert=ASC(MID$(spiele$(spiele),k,1))-48
  IF wert THEN
   IF wert=6 THEN
    PUT (j,i),mauer%
   END IF
   IF wert=2 THEN
    PUT (j,i),ziel%:ziele=ziele+1
   END IF
   IF wert=7 THEN
    PUT (j,i),kugel%
   END IF
   IF wert=4 THEN
    PUT (j,i),face%:posx=j:posy=i
   END IF
  END IF
  k=k+1
  NEXT j
 NEXT i
 setflag=1:COLOR 12
 x1=159:x2=456:y1=242:y2=260
 LINE (x1,y1)-(x2,y2),3,b
 GOSUB fenster
 LOCATE 32,22:PRINT "Spiel";spiele
 LOCATE 32,32:PRINT "Zuege";zug
 LOCATE 32,43:PRINT "Schritte";schritt
 POKEW 14676118&,33664&
 RETURN
wartel:
 FOR i=1 TO 500
 NEXT i
 RETURN
violation:
 PRINT "Es ist folgender Fehler aufgetreten!
 Nr:";ffehler
 END
REM ********* Ende von HARD-WORK ********************
```

Listing 1. HARD-WORK

```
REM ***************************************************
REM *               HARD-WORK Druckerfile             *
REM ***************************************************

DIM spiele$(20)
OPEN "Dat" FOR INPUT AS#1
 INPUT #1,maxa
 FOR i=1 TO maxa:LINE INPUT #1,spiele$(i):NEXT i
 CLOSE 1
start:CLS
 PRINT "Es sind";maxa;"Spiele vorhanden"
 INPUT "Welches Spiel";sp
 INPUT "Auf Drucker oder Schirm";s$
 IF s$="d" THEN drucken
zeichnen:
 CLS:k=1
 FOR i=1 TO 15:FOR j=1 TO 20
 wert=ASC(MID$(spiele$(sp),k,1))-48
 IF wert=0 THEN LOCATE i,j:PRINT"-"
 IF wert=6 THEN LOCATE i,j:PRINT "#"
 IF wert=7 THEN LOCATE i,j:PRINT "o"
 IF wert=2 THEN LOCATE i,j:COLOR 2:PRINT "+":COLOR 1
 IF wert=4 THEN LOCATE i,j:COLOR 3:PRINT "":COLOR 1
 k=k+1:NEXT j:NEXT i:PRINT "Spiel";sp,"  <SPACE>"
warte:a$=INPUT$(1):IF a$=CHR$(32) THEN start
 GOTO warte
drucken:k=1:l=20
 LPRINT "Spiel";sp
 FOR y=1 TO 15:LPRINT MID$(spiele$(sp),k,l):k=k+20
 NEXT y
 GOTO start
REM ********* Ende HARD-WORK Druckerfile *************
```

Listing 2. HARD-WORK Druckerfile

```
REM ***************************************************
REM *             HARD-WORK Spielegenerator           *
REM ***************************************************

a$(1)="000000000000000000000000000000000000000000000
000000000000000"
a$(2)="000000000000000000000000000000000000000000000
000000000000000"
a$(3)="666066666060666066606060606060606400606066606
060606066606660"
a$(4)="606060606060606060606262626262626662626207070
707070700070007"
a$(5)="000000000000000000000000000000000000070000000
000000000000000"
gen$(1)=a$(1)+a$(2)+a$(3)+a$(4)+a$(5)
a$(1)="606066606660662000006060606060606060000066606
660666060606660"
a$(2)="606060606600606000006060606060606600000000070
700020007000000"
a$(3)="000020007000200000060606066606660606000062606
060606060660000"
a$(4)="676060606066606600006460606060660060600066666
066606060606000"
a$(5)="000000000000000000000000000000000000000000000
000000000000000"
gen$(2)=a$(1)+a$(2)+a$(3)+a$(4)+a$(5)
a$(1)="666666000000000000000622006666666666600000622 00
600000000600000"
a$(2)="622006707006706000006220407067070066666662200
670707000007006"
a$(3)="622666000000670006066622607000767000700066226 0
077606007606666"
a$(4)="622600607067006060006266000000007060600062600
766666666606000"
a$(5)="626000000000000600062666666666666666600066600
000000000000000"
gen$(3)=a$(1)+a$(2)+a$(3)+a$(4)+a$(5)
OPEN "Dat"FOR OUTPUT AS#1
 PRINT #1,3
 FOR i=1 TO 3:PRINT #1,gen$(i):NEXT i
CLOSE 1
SYSTEM
REM ******** Ende von HARD-WORK Spielegenerator ******
```

Listing 3. HARD-WORK Spielegenerator

Franz-Josef Reichert

Vielseitige Requester

Grundlagen der Library-Programmierung in C

Verfolgen Sie, wie wir Stück für Stück eine komplette Amiga-Library zusammenstellen. Für Ihre eigenen Programme stehen Ihnen ab sofort jederzeit drei flexible Unterroutinen zur Verfügung: ein leistungsfähiger File-Requester, ein komfortables Farbeinstellungsfenster für maximal 32 Farben und eine Standard-Dialogbox für beliebigen Text und bis zu drei Entscheidungs-Gadgets. Sie sind mit unserer Library an keine bestimmte Programmiersprache gebunden und haben damit für oft gestellte Programmieraufgaben immer eine passende Lösung zur Hand.

Bedienungskomfort beim Amiga fängt mit Intuition an. Intuition als Betriebssystemteil stellt für Anwenderprogramme die nötigen Funktionen zur Verfügung, um ein ansprechendes Benutzer-Interface aufzubauen. Begriffe wie Menüs, Gadgets, Windows und Requester sollten für den Amiga-verwöhnten Computeristen längst keine Fremdwörter mehr sein (vergleiche [1]). Ihre Benutzerfreundlichkeit stellen diese Elemente in unzähligen Programmen eindrucksvoll unter Beweis. Aber: Was des einen Freud, ist des anderen Leid..., so oder ähnlich hat sicher schon mancher Programmierer gestöhnt, der ein Programm entwickeln und mit einem solchen hohen Bedienungskomfort ausstatten wollte. Denn all das, was die Bedienung so schön einfach macht, ist unvergleichlich schwieriger zu programmieren. Schon oft ist ein gutes Projekt gescheitert, weil sich im nachhinein die Benutzerschnittstelle als zu unhandlich erwiesen hat. Um dies von vornherein zu vermeiden, muß schon bei der Programmkonzeption und Planung ein erheblicher Teil der Entwicklungszeit für die Schaffung eines komfortablen Bedienungskonzeptes veranschlagt werden. Wir wollen Ihnen einen Teil dieser Arbeit abnehmen und Ihnen in der heutigen und den folgenden Ausgaben der AMIGA DOS insgesamt drei nützliche C-Funktionen vorstellen, die Sie diese Sorgen vergessen lassen und es Ihnen ermöglichen, sich guten Gewissens auf das eigentliche Problem Ihres Programmierprojektes zu konzentrieren.

Unsere Routinensammlung wird in einer »shared-library« zur Verfügung stehen, so daß sie von allen Programmiersprachen aus genutzt werden kann, egal, ob Sie in C, Assembler, BASIC, Modula2 oder was auch immer programmieren. Die von Ihnen verwendete Programmiersprache spielt überhaupt keine Rolle, sie muß nur eine Möglichkeit bieten, Funktionen aus Amiga-Libraries aufzurufen. Die komplette Library können Sie auf der DATA-BOX in fertig compilierter und gelinkter Form beziehen. Wer die komplette Artikelserie verfolgt, wird aber nicht nur eine Menge über Intuition lernen, sondern am Ende auch ein grundlegendes Konzept der Amiga-Philosophie verstehen und anwenden können: die »shared-library« als wesentlicher Bestandteil der offenen Systemarchitektur. Eigene Erweiterungen des Betriebssystems sind damit leicht, problemlos und systemkonform zu realisieren (vergleiche [2], A, G-13ff.; [3] S. 75ff.).

Was bedeutet eigentlich »shared-library«? Nicht nur ein Programm kann die Routinen benutzen, sondern sogar mehrere zugleich, sie teilen sich quasi die Library. Gerade für eine Multitasking-Maschine wie den Amiga bietet dieses Konzept unschätzbare Vorteile. Mehrere Programme verwenden die gleichen Routinen zur Erzeugung der Bedienungselemente, die unsere Library aber nur einmal zur Verfügung stellen muß. Das spart zum einen Speicher, zum anderen sorgt es für eine einheitliche Benutzerführung. Mehr dazu lesen Sie im dritten Teil unserer Artikelserie. Einen Überblick über unsere Werkstatt-Serie finden Sie auf der nächsten Seite.

Schnelle Entscheidung ist gefragt – Der File-Requester

In praktisch jedem Programm ist es früher oder später erforderlich, daß durch den Benutzer ein Bezeichnername für eine Datei eingegeben wird. Auf diese Weise werden zum Beispiel Daten unter diesem Namen auf einem Massenspeicher gesichert oder von diesem geladen. Die einfachste Möglichkeit der Abfrage wäre ein simpler Eingabeprompt mit der Aufforderung "Bitte Dateinamen eingeben: ?" und nachfolgender Tastatureingabe durch den Benutzer. Diese Art der Benutzerführung wird aber keinesfalls den Fähigkeiten eines Amiga gerecht. Intuition ermöglicht es, durch Requester und Gadgets eine angenehmere Art der Kommunikation zwischen Benutzer und Programm zu schaffen. Gemeint sind damit File-Requester, wie sie von professionellen Programmen her bekannt sind. Diese fragen nicht nur einfach eine Zeichenkette ab, sondern geben gleichzeitig einen Überblick über die verfügbaren Disketten- und Festplattenlaufwerke, den darauf vorhandenen Directories und darin enthaltenen Files. Durch Gadgets kann sich der Benutzer bequem in der Dateihierarchie bewegen und den geforderten Dateinamen einfach mit der Maus auswählen oder über einen String-Gadget eingeben. Fehlbedienungen werden so weitestgehend vermieden. Da jedoch File-Requester für die unterschiedlichsten Programme ähnlich arbeiten und aussehen, liegt nichts näher; als einen Standard-File-Requester zu entwickeln und diesen in allen Programmen wieder zu verwenden. Auch für den Benutzer bringt es erhebliche Vorteile, in jedem Programm wieder auf vertraute Bedienungselemente zu treffen, deren Gebrauch nicht erst neu erlernt werden muß. Mit der vorliegenden Library-Routine »BuildFile-Requester()« steht Ihnen für Ihre eigenen Anwendungen ein besonders komfortabler Vertreter dieser Gattung zur Verfügung. Nach dem Aufruf aus einem beliebigen Programm erscheint in Ihrem Window ein Requester, der die zur Verfügung stehenden Laufwerke auf der linken Seite in Form von Piktogrammen anzeigt. Dies können Disketten- oder Festplattenlaufwerke sein, aber auch virtuelle Laufwerke, wie zum Beispiel die RAM-Disk, eben alle DOS-Devices, die der Amiga standardmäßig oder optional zur Verfügung stellt. Bis zu fünf Devices finden hier Platz, was in den allermeisten Fällen ausreichen wird. Durch einfaches Anklicken mit der Maus wird das Inhaltsverzeichnis des entsprechenden Devices gelesen und im Requester angezeigt. Dazu stehen auf der rechten Seite fünf Auswahlfelder für die Directories und acht für die Files zur Verfügung, in denen die Namen alphabetisch geordnet erscheinen. Sind mehr Einträge vorhanden, als in den Feldern angezeigt werden können, lassen sich die nicht sichtbaren Einträge durch Scrollen der Auswahlfelder erreichen. Dies geschieht in bekannter Manier durch Proportional-Gadgets am rechten Rand der Auswahlfelder. Eine Dauerbetätigung der Pfeil-Gadgets führt dabei zum kontinuierlichen Scrollen. Die Rollbalken können sowohl durch direktes Anklicken und Verschieben als auch durch Anklicken der Freiräume in der Auswahlbox in die entsprechende Position gebracht werden. Um in der Dateistruktur ein Directory abzusteigen, wird das gewünschte Directory in den oberen Auswahlfeldern einfach mit der Maus angeklickt. Sofort werden die Einträge gelesen und die Auswahlfelder entsprechend aktualisiert. Haben Sie sich in der Auswahl vertan, so brauchen Sie übrigens nicht zu warten, bis das Directory komplett gelesen ist. Betätigen Sie einfach das richtige Gadget, und die Routine fährt unverzüglich mit dem Lesen des neuen Directorys fort. Bei all diesen Operationen erscheint der korrekte Pfadname im ersten String-Gadget am unteren Rand des Requesters.

Schnellentschlossene können hier auch direkt den gewünschten Verzeichnisnamen über die Tastatur ein-

geben. Dabei ist auch die Angabe von logischen Gerätenamen, wie »devs:« oder »libs:«, aber auch allen mit »assign« zugeordneten Namen möglich. In der linken unteren Ecke befinden sich noch drei Boolean-Gadgets. Mit »OK« wird der gewählte Dateiname mit dem zugehörigen Pfad an das aufrufende Programm zurückgegeben, »CANCEL« dagegen bricht den Request mit einem Null-String als Resultat ab. Mit »PARENT« kann in das übergeordnete Directory zurückgegangen werden. Im untersten String-Gadget wird der Dateiname angezeigt und kann gegebenenfalls geändert werden. Wird hier [RETURN] betätigt, wird ebenfalls der Request mit Pfad- und Dateiname als Resultat beendet.

Im Anschluß finden Sie alle benötigten Listings für den Lattice-C-Compiler V5.04. »BuildFileRequester.c« (Listing 1) umfaßt den kompletten Quellcode unserer Funktion. Zusätzlich benötigen Sie noch die Definition der Grafikelemente, die in »FileRequesterImages.c« (Listing 2) abgelegt sind. Zwei Funktionen zur Umrandungsdefinition und Textzentrierung liefert »global.c« (Listing 3). Sie wurden aus dem Haupttext ausgegliedert, weil sie in der fertigen Library auch aus den beiden anderen Library-Funktionen heraus benutzt werden sollen, die wir Ihnen in den folgenden Teilen vorstellen werden. Schließlich schicken wir heute bereits das Header-File »requester.h« (Listing 4) voraus, das besonders für die C-Programmierer zur Benutzung der fertigen Library von Bedeutung ist. Momentan liefert es uns nur die Funktionsprototypen für unsere drei Library-Routinen.

Damit haben Sie im Grunde alles beisammen, um bis zur nächsten Folge ein kleines Beispielprogramm in C zu entwickeln. Sie brauchen nur ein Intuition-Window zu öffnen und die Funktion »BuildFileRequester()« aufzurufen. Insgesamt sind dazu fünf Parameter erforderlich:

```
name =
BuildFileRequester(dirname,no
dename,filename,window,title)
Parameter:
        UBYTE *dirname,*noden
ame,*filename;
        struct Window *windo
w;
        UBYTE *title;
Rückgabewert:
        UBYTE *name;
```

Die Bedeutung der Parameter im einzelnen:

»dirname« – Ein Zeiger auf den Pfadnamen-Puffer; dieser Puffer sollte mindestens 128 Bytes (einschließlich Nullbyte) umfassen. Hier kann auch ein voreingestellter Pfadname übergeben werden; das so bezeichnete Directory wird unmittelbar nach dem Aufruf eingelesen. Wird ein leerer (das heißt Null-) String übergeben, so wird mit dem Ursprungs-Directory (»:«) begonnen.

»nodename« – Ein Zeiger auf den Dateinamen-Puffer; dieser Puffer sollte mindestens 32 Bytes (einschließlich Nullbyte) umfassen. Auch hier kann ein Initialstring übergeben werden.

»filename« – Ein Zeiger auf den Puffer für den Rückgabewert; dieser Puffer wird auch intern von der Routine genutzt und sollte mindestens 256 Zeichen (einschließlich Nullbyte) umfassen.

»window« – Ein Zeiger auf die Intuition-Window-Struktur des Fensters, in dem der Requester erscheinen soll.

»title« – Ein Zeiger auf einen Kopfzeilen-String, der als Titelzeile im Requester erscheinen soll. Dieser String wird automatisch zentriert und darf maximal 32 Zeichen (ohne Nullbyte) umfassen, um ganz sichtbar zu sein.

Die Bedeutung des Rückgabewertes:

»name = = -1« – Während des Aufrufs der Routine trat ein Fehler auf. Dies kann seinen Grund in mangelndem Speicherplatz oder falschen Aufrufparametern haben.

»name = = 0« – Der Benutzer hat den Requester mit Betätigung des Cancel-Gadget beendet.

»name > 0« – Der Benutzer hat den Requester mit Betätigung des unteren String-Gadgets oder des OK-Gadgets beendet; »name« zeigt auf den Filenamen (mit Pfadnamen), der das Ergebnis des Requests darstellt.

Beim Linken müssen Sie die drei Objektmodule »BuildFileRequester.o«, »global.o« und »FileRequesterImages.o« angeben. Sie erhalten diese durch folgende Aufrufe:

```
»lc -v -w -O BuildFileRequest
er global FileRequesterImages
```

Ein fertiges (minimales) Beispielprogramm mit Namen »minex.c« haben wir in Listing 5 vorgesehen. Compilieren und linken Sie es mit den Aufrufen:

```
lc -v -w -O -Lcn+buildfilereq
uester.o+global.o+ file-
requesterimages.o minex
```

Beim Aufruf des Programms öffnet sich ein Window auf dem Workbench-Screen. Durch Betätigen der linken Maustaste erscheint darin der File-Requester. Rückgabewerte der Funktion werden im »Console-Window« protokolliert.

Damit verfügen Sie bereits heute über die Möglichkeit, Ihre C-Programme mit einem leistungsfähigen File-Requester auszustatten. Gegenstand der nächsten Folge wird die Dialogbox und das Farbeinstellungsfenster sein.

Hier noch Hinweise zur Literatur:

[1] Mical, R.J., Deyl, S.: Amiga Intuition Reference Manual; 4. Auflage, Addison-Wesley 1987, ISBN 0-201-11076-8.

[2] Commodore-Amiga, Inc.: Amiga ROM Kernel Reference Manual: Includes and Autodocs, revised and updated; 1. Auflage, Addison-Wesley 1989, ISBN 0-201-18177-0.

[3] Commodore-Amiga, Inc.: Amiga ROM Kernel Reference Manual: Exec; 4. Auflage, Addison-Wesley 1986, ISBN 0-201-11099-7.

(jb)

Eine kleine Übersicht

Unser mehrteiliger Werkstatt-Artikel beschäftigt sich mit der Erstellung von eigenen Libraries, die zwar in der Sprache C programmiert wurden, aber auf alle gängigen Amiga-Computersprachen umgesetzt werden können. Mehrere Programme, wie der hier abgedruckte File-Requester, werden Ihnen dabei als "Unterlage" zur Verfügung gestellt. In den Beiträgen finden Sie folgende Themen:

- Überblick, Beschreibung und Listings des File-Requesters
- Beschreibung und Listings zu Auswahlbox und Colorpalette
- Wissenswertes zu Libraries, Listing des Library-Rumpfes, Zusammenbau aller Routinen als Amiga-Library
- Anwendungsbeispiele, Linker-Library, Aufruf aus allen Programmiersprachen

Listings

```
/*==================================================*
*==                                              ==*
*==      BuildFileRequester.C                    ==*
*==      Programmed  by  Franz-Josef  Reichert   ==*
*==      Lattice C - Compiler V 5.04             ==*
*==                                              ==*
*==================================================*/

#include <clib/macros.h>
#include <exec/types.h>
#include <exec/memory.h>
#include <intuition/intuition.h>
#include <libraries/dosextens.h>
#include <proto/exec.h>
#include <proto/dos.h>
```

Listing: Requester-Library

```
#include <proto/intuition.h>
#include <string.h>
#include "requester.h"

#define STRSIZE     128
#define DISPSIZE    19
#define REQWIDTH    268
#define REQHEIGHT   194
#define MAXGADGETS  29
#define MAXDEVS     5
#define MAXNAME     32
#define DEVNAMELEN  8
#define ET_DIR      1
#define ET_FILE     -1
#define MAXDIR      5
#define MAXFILE     8

extern UWORD __chip floppy[],__chip selfloppy[];
```



Listing: Requester-Library

```
extern UWORD __chip arrowup[],__chip arrowdown[];
extern struct TextAttr taPlain;
extern struct DosLibrary *DOSBase;
static UBYTE *const BGTxt[] =
{ "OK","PARENT","CANCEL" };
extern VOID CenterIText
(struct IntuiText*,WORD,WORD,WORD,WORD);
extern VOID InitBorder(WORD*,WORD,WORD,WORD,WORD);

static struct HandlerMsg {
 struct StandardPacket  hm_Pkt;
 struct InfoData        hm_Info;
};

static struct Buffer {
 struct Buffer *succ;
 BYTE          type;
 UBYTE         *name,namebuffer[MAXNAME];
};

static struct DATA {
 struct Border GadgBorder,StrBorder;
 struct Border ReqBorder,DirBorder,FileBorder;
 struct Buffer *firstbuf;
 struct Buffer *lastbuf;
 struct Gadget *Gadg[MAXGADGETS];
 struct Image ArrowupImg,ArrowdownImg;
 struct Image dummyImg[2];
 struct Image FloppyImg,SelFloppyImg;
 struct IntuiText *GadgText[MAXGADGETS - 8];
 struct IntuiText headertxt[2];
 struct PropInfo PrInfo[2];
 struct Remember *Rkey;
 struct Requester filereq;
 struct StringInfo StrInfo[2];
 struct Window *dispwindow;
 LONG dcount,fcount,bufcount;
 WORD doff,foff;
 WORD Rvec[16],Dvec[16],Fvec[16],Gvec[16],Svec[16];
 UBYTE *dirbuf,*filebuf,*undobuf;
 UBYTE *namelist[MAXDEVS+1];
 UBYTE devnames[MAXDEVS][DEVNAMELEN];
 UBYTE namebuf[MAXDIR+MAXFILE][DISPSIZE];
};

static LONG __regargs ScrollField
(struct Buffer*,WORD,struct DATA*);
static struct Buffer __regargs *GetBuf
(UWORD,struct DATA*);
static struct Buffer __regargs *GetDir
(UBYTE*,ULONG,struct DATA*);
static struct Gadget __regargs *AllocGadgets
(UBYTE**,UWORD,UBYTE*,struct DATA*);
static struct DATA __regargs *InitGlobal(VOID);
static UBYTE __regargs *BldPath(BPTR,UBYTE*);
static ULONG __regargs GetData(struct Buffer*,
WORD,WORD,UBYTE,UBYTE,struct DATA*);
static VOID __regargs BubbleSort(struct Buffer*);
static VOID __regargs FillBlank(UBYTE*);
static VOID __regargs FindDevices(struct DATA*);
static VOID __regargs InitGadget
(WORD,WORD,WORD,WORD,WORD,struct DATA*);
static VOID __regargs RemakeGadgs
(struct Gadget*,struct Gadget*,struct DATA*);
static WORD __regargs ReadProp
(ULONG,UBYTE,UBYTE,struct DATA*);
static ULONG __regargs DiskInfo(struct MsgPort*);

static struct DATA __regargs *InitGlobal()
{
 struct DATA *gd;
 if(gd = (struct DATA*)AllocMem(sizeof
 (struct DATA),MEMF_PUBLIC|MEMF_CLEAR)) {
  gd->headertxt[0].FrontPen = 3;
  gd->headertxt[0].ITextFont =
  gd->headertxt[1].ITextFont = &taPlain;
  gd->headertxt[1].NextText = &gd->headertxt[0];
  gd->ReqBorder.FrontPen =
  gd->DirBorder.FrontPen = gd->FileBorder.FrontPen =
  gd->StrBorder.FrontPen = 1;
  gd->GadgBorder.FrontPen = 3;
  gd->ReqBorder.Count = gd->DirBorder.Count =
  gd->FileBorder.Count = gd->GadgBorder.Count =
  gd->StrBorder.Count = 8;
  gd->ReqBorder.XY  = gd->Rvec;
  gd->DirBorder.XY  = gd->Dvec;
  gd->FileBorder.XY = gd->Fvec;
  gd->GadgBorder.XY = gd->Gvec;
  gd->StrBorder.XY  = gd->Svec;
  gd->FileBorder.NextBorder = &gd->DirBorder;
  gd->DirBorder.NextBorder = &gd->ReqBorder;
  gd->FloppyImg.Width = gd->SelFloppyImg.Width = 67;
  gd->FloppyImg.Height = gd->SelFloppyImg.Height = 12;
  gd->FloppyImg.Depth = gd->SelFloppyImg.Depth = 2;
  gd->FloppyImg.PlanePick =
  gd->SelFloppyImg.PlanePick = 0x3;
  gd->ArrowupImg.LeftEdge =
  gd->ArrowdownImg.LeftEdge = -2;
  gd->ArrowupImg.TopEdge = -1;
  gd->ArrowupImg.Width = gd->ArrowdownImg.Width = 17;
  gd->ArrowupImg.Height = gd->ArrowdownImg.Height = 11;
  gd->ArrowupImg.Depth = gd->ArrowdownImg.Depth = 1;
  gd->ArrowupImg.PlanePick =
  gd->ArrowdownImg.PlanePick = 1;
```

Listing: Requester-Library

```
  gd->FloppyImg.ImageData = floppy;
  gd->SelFloppyImg.ImageData = selfloppy;
  gd->ArrowupImg.ImageData = arrowup;
  gd->ArrowdownImg.ImageData = arrowdown;
 }
 return(gd);
}

static struct Buffer __regargs *GetDir
(UBYTE *name,ULONG sigmask,struct DATA *gd) {
 BPTR  lock;
 UWORD entries = 0;
 struct Buffer *buf = NULL;
 struct FileInfoBlock *fib;
 SetSignal(NULL,sigmask);
 if(fib = (struct FileInfoBlock*)AllocMem(sizeof
 (struct FileInfoBlock),MEMF_PUBLIC|MEMF_CLEAR)) {
  if(name[0] != NULL && (lock = Lock
  (name,ACCESS_READ))) {
   gd->doff = gd->foff = NULL;
   Examine(lock,fib);
   while(ExNext(lock,fib)) {
    entries++;
    if((buf = GetBuf(entries,gd)) == NULL) break;
    strcpy(buf->namebuffer,fib->fib_FileName);
    buf->name = buf->namebuffer;
    buf->type = (fib->fib_DirEntryType > 0) ?
    (ET_DIR) : (ET_FILE);
    if(buf->succ) buf->succ->name = NULL;
    buf = gd->firstbuf;
    if(sigmask & SetSignal(NULL,sigmask)) break;
   }
   UnLock(lock);
   BubbleSort(buf);
  }
  FreeMem((UBYTE*)fib,sizeof(struct FileInfoBlock));
  gd->dcount = ScrollField(buf,ET_DIR,gd);
  gd->fcount = ScrollField(buf,ET_FILE,gd);
 }
 gd->lastbuf = NULL;
 return(buf);
}

static struct Buffer __regargs *GetBuf
(UWORD num,struct DATA *gd) {
 struct Buffer *buf;
 if(num > gd->bufcount) {
  if(buf = (struct Buffer*)AllocRemember
  (&gd->Rkey,sizeof(struct Buffer),
  MEMF_PUBLIC|MEMF_CLEAR)) {
   gd->bufcount++;
   if(gd->firstbuf == NULL) gd->firstbuf = buf;
   else gd->lastbuf->succ = buf;
  } else return(NULL);
 } else buf = ((gd->lastbuf) ?
 (gd->lastbuf->succ) : (gd->firstbuf));
 gd->lastbuf = buf;
 return(buf);
}

static VOID __regargs BubbleSort(struct Buffer *b) {
 struct Buffer *block;
 UBYTE *name,swap;
 BYTE type;
 do {
  swap = FALSE;
  block = b;
  while(block && block->succ && block->succ->name) {
   if(stricmp(block->name,block->succ->name) > 0) {
    name = block->name;
    block->name = block->succ->name;
    block->succ->name = name;
    type = block->type;
    block->type = block->succ->type;
    block->succ->type = type;
    swap = TRUE;
   }
   block = block->succ;
  }
 } while(swap == TRUE);
 return;
}

static VOID __regargs FillBlank(UBYTE *string) {
 while(strlen(string) < DISPSIZE - 1)
  strcat(string," ");
 return;
}

static ULONG __regargs GetData
(struct Buffer *buffer,WORD offset,WORD type,
UBYTE pos,UBYTE max,struct DATA *gd) {
 ULONG count = 0;
 while(buffer && buffer->name) {
  if(buffer->type == type) {
   if(offset <= 0 && pos <= max) {
    stccpy(gd->GadgText[pos]->IText,
    buffer->name,DISPSIZE);
    gd->Gadg[pos]->UserData = (APTR)buffer->name;
    if(type > 0) {
     strncat(gd->GadgText[pos]->IText," (dir)",
     DISPSIZE - 1 - strlen(gd->GadgText[pos]->IText));
    }
    FillBlank(gd->GadgText[pos]->IText);
```



Listing: Requester-Library

```
    pos++;
   }
   count++;
   offset--;
  }
  buffer = buffer->succ;
 }
 while(pos <= max) {
  gd->GadgText[pos]->IText[0] = NULL;
  FillBlank(gd->GadgText[pos]->IText);
  gd->Gadg[pos]->UserData = NULL;
  pos++;
 }
 return(count);
}

static VOID __regargs RemakeGadgs(struct Gadget *first,
struct Gadget *last,struct DATA *gd) {
 ULONG gadgcount;
 struct Gadget *gadg;
 for(gadgcount = 1, gadg = first; gadg != last
 && gadg; gadg = gadg->NextGadget, gadgcount++);
 RefreshGList(first,gd->dispwindow,
 &gd->filereq,gadgcount);
 return;
}

static ULONG __regargs DiskInfo(struct MsgPort *proc) {
 struct HandlerMsg *hm;
 struct MsgPort *reply;
 ULONG bpb = 0;
 if(hm = (struct HandlerMsg*)AllocMem
 (sizeof(struct HandlerMsg),MEMF_PUBLIC¦MEMF_CLEAR)) {
  if(reply = CreatePort(NULL,0)) {
   hm->hm_Pkt.sp_Msg.mn_Node.ln_Name =
   (UBYTE*)&hm->hm_Pkt.sp_Pkt;
   hm->hm_Pkt.sp_Pkt.dp_Link = &hm->hm_Pkt.sp_Msg;
   hm->hm_Pkt.sp_Pkt.dp_Port = reply;
   hm->hm_Pkt.sp_Pkt.dp_Type = ACTION_DISK_INFO;
   hm->hm_Pkt.sp_Pkt.dp_Arg1 =
   (ULONG)&hm->hm_Info >>á2;
   PutMsg(proc,&hm->hm_Pkt.sp_Msg);
   WaitPort(reply);
   GetMsg(reply);
   if(hm->hm_Pkt.sp_Pkt.dp_Res1 == DOSTRUE)
    bpb = hm->hm_Info.id_BytesPerBlock;
   DeletePort(reply);
  }
  FreeMem((UBYTE*)hm,sizeof(struct HandlerMsg));
 }
 return(bpb);
}

static VOID __regargs FindDevices (struct DATA *gd) {
 struct DevInfo *devinfo;
 UWORD i = 0; UBYTE *dname;
 Forbid();
 devinfo = (struct DevInfo*)BADDR(((struct DosInfo*)
 BADDR(((struct RootNode*)DOSBase->dl_Root)->
 rn_Info))->di_DevInfo);
 while(devinfo && i < MAXDEVS) {
  if(devinfo->dvi_Type == DLT_DEVICE
  && devinfo->dvi_Task && DiskInfo
  ((struct MsgPort*)devinfo->dvi_Task)) {
   dname = (UBYTE*)BADDR(devinfo->dvi_Name);
   gd->namelist[i] = (UBYTE*)&gd->devnames[i];
   stccpy(gd->namelist[i],dname + 1,
   MIN(*dname + 1,DEVNAMELEN - 1));
   strcat(gd->namelist[i],":"); i++;
  }
  devinfo = (struct DevInfo*)BADDR(devinfo->dvi_Next);
 }
 Permit();
 return;
}

static UBYTE __regargs *BldPath(BPTR lock,UBYTE *buf) {
 struct FileInfoBlock *fib;
 BPTR newlock;
 UBYTE *namepos,*dname,
 diskname[MAXNAME],pathname[STRSIZE];
 *pathname = NULL;
 if(fib = (struct FileInfoBlock*)AllocMem(sizeof
 (struct FileInfoBlock),MEMF_PUBLIC¦MEMF_CLEAR)) {
  dname = (UBYTE*)BADDR(((struct DeviceList*)
  BADDR(((struct FileLock*)BADDR(lock))->
  fl_Volume))->dl_Name);
  stccpy(diskname,dname + 1,*dname + 1);
  while(lock) {
   if(Examine(lock,fib)) {
    if(fib->fib_DirEntryType > 0) {
     strins(pathname,"/");
     strins(pathname,fib->fib_FileName);
    }
    else {
     UnLock(lock);
     *pathname = NULL;
     break;
    }
   }
   newlock = ParentDir(lock);
   UnLock(lock);
   lock = newlock;
  }
```



Listing: Requester-Library

```
  *(pathname + strlen(pathname) - 1) = ':';
  for(namepos = pathname; *namepos; namepos++) {
   if(*namepos == '/') {
    *namepos = ':';
    *(pathname + strlen(pathname) - 1) = NULL;
    break;
   }
  }
  if(*pathname == ':') strins(pathname,diskname);
  FreeMem((BYTE*)fib,sizeof(struct FileInfoBlock));
 }
 else *pathname = NULL;
 strcpy(buf,pathname);
 return(buf);
}

static WORD __regargs ReadProp(ULONG entries,
UBYTE pgadg,UBYTE count,struct DATA *gd) {
 WORD offset = (((struct PropInfo*)
 gd->Gadg[pgadg]->SpecialInfo)->VertPot
 * (entries - count + 1)) >> 16;
 return((WORD)(MAX(offset,0)));
}

static LONG __regargs ScrollField(struct Buffer *buf,
WORD type,struct DATA *gd) {
 WORD  offset = ((type > 0) ? (gd->doff):(gd->foff));
 UBYTE fgadg  = ((type > 0) ? ( 3):( 8));
 UBYTE lgadg  = ((type > 0) ? ( 7):(15));
 UBYTE pgadg  = ((type > 0) ? (22):(23));
 UBYTE count  = lgadg + 1 - fgadg;
 LONG ents   = GetData(buf,offset,type,fgadg,lgadg,gd);
 LONG vbody = (LONG)count * 0xFFFFL / MAX(ents,count);
 LONG vpot  = ((LONG)offset * (0xFFFFFFFF /
 ((ents - count) ? (ents - count) : (1L)))) >> 16;
 NewModifyProp(gd->Gadg[pgadg],gd->dispwindow,&gd->
 filereq,AUTOKNOB¦FREEVERT,NULL,vpot,NULL,vbody,1L);
 RemakeGadgs(gd->Gadg[fgadg],gd->Gadg[lgadg],gd);
 return(ents);
}

static VOID __regargs InitGadget(WORD num,WORD left,
WORD top,WORD width,WORD height,struct DATA *gd) {
 gd->Gadg[num]->LeftEdge    = left;
 gd->Gadg[num]->TopEdge     = top;
 gd->Gadg[num]->Width       = width;
 gd->Gadg[num]->Height      = height;
 gd->Gadg[num]->NextGadget  = gd->Gadg[num + 1];
 gd->Gadg[num]->Activation ¦= RELVERIFY;
 gd->Gadg[num]->GadgetType ¦= REQGADGET;
}

static struct Gadget __regargs *AllocGadgets
(UBYTE *dl[],UWORD fheight,UBYTE *text,
struct DATA *gd) {
 WORD i;
 struct Gadget *last;
 for(i = 0; i < MAXGADGETS; i++) {
  if((gd->Gadg[i] = (struct Gadget*)AllocRemember
  (&gd->Rkey,sizeof(struct Gadget),
  MEMF_PUBLIC¦MEMF_CLEAR)) == NULL) return(NULL);
  gd->Gadg[i]->GadgetID = (UWORD)i;
 }
 for(i = 0; i < MAXGADGETS - 8; i++) {
  if((gd->GadgText[i] = (struct IntuiText*)
  AllocRemember(&gd->Rkey,sizeof(struct IntuiText),
  MEMF_PUBLIC¦MEMF_CLEAR)) == NULL) return(NULL);
  if(i > 2 && i < 16) {
   gd->GadgText[i]->IText = (UBYTE*)&gd->namebuf[i-3];
   gd->GadgText[i]->LeftEdge = 2;
   gd->GadgText[i]->TopEdge  = 1;
   gd->GadgText[i]->BackPen  = 0;
   gd->GadgText[i]->DrawMode = JAM2;
  }
  else gd->GadgText[i]->DrawMode = JAM1;
  gd->GadgText[i]->FrontPen  = 1;
  gd->GadgText[i]->ITextFont = &taPlain;
 }
 for(i = 0; i < 3; i++) {
  InitGadget(i,13,(WORD)(143 + i * 15),63,11,gd);
  gd->Gadg[i]->GadgetType  ¦= BOOLGADGET;
  gd->Gadg[i]->GadgetRender = (APTR)&gd->GadgBorder;
  gd->Gadg[i]->GadgetText = gd->GadgText[i];
  gd->GadgText[i]->IText = BGTxt[i];
  if(i != 1) gd->Gadg[i]->Activation ¦= ENDGADGET;
  CenterIText(gd->GadgText[i],1,1,
  gd->Gadg[i]->Width,gd->Gadg[i]->Height);
 }
 InitBorder(gd->Gvec,-2,-1,63,11);
 for(i = 3; i < 16; i++) {
  InitGadget(i,89,(WORD)(20 + (i - 3) * 9),147,10,gd);
  gd->Gadg[i]->GadgetType ¦= BOOLGADGET;
  gd->Gadg[i]->GadgetText  = gd->GadgText[i];
 }
 for(i = 8; i < 16; i++)
  InitGadget(i,89,(WORD)(76 + (i - 8) * 9),147,10,gd);
 for(i   16; i < 20; i++) {
  gd->Gadg[i]->GadgetType ¦= BOOLGADGET;
  gd->Gadg[i]->Flags  ¦= GADGIMAGE;
  gd->Gadg[i]->Activation  = GADGIMMEDIATE;
 }
 gd->Gadg[16]->GadgetRender = gd->Gadg[18]->
 GadgetRender = (APTR)&gd->ArrowdownImg;
 gd->Gadg[17]->GadgetRender = gd->Gadg[19]->
```

Listing: Requester-Library

```
 GadgetRender = (APTR)&gd->ArrowupImg;
 InitGadget(16,239,139,13,10,gd);
 InitGadget(17,239, 76,13,10,gd);
 InitGadget(18,239, 56,13,10,gd);
 InitGadget(19,239, 20,13,10,gd);
 for(i = 20; i < 22; i++) {
  gd->Gadg[i]->GadgetType   |= STRGADGET;
  gd->Gadg[i]->GadgetRender = (APTR)&gd->StrBorder;
  gd->Gadg[i]->SpecialInfo  = (APTR)&gd->StrInfo[i-20];
 }
 InitGadget(20,90,158,167,(WORD)(fheight + 2),gd);
 InitGadget(21,90,174,167,(WORD)(fheight + 2),gd);
 gd->Gadg[21]->Activation |= ENDGADGET;
 InitBorder(gd->Svec,-5,-2,163,(WORD)(fheight + 1));
 for(i = 22; i < 24; i++) {
  gd->Gadg[i]->GadgetType   |= PROPGADGET;
  gd->Gadg[i]->Flags        |= GADGIMAGE;
  gd->Gadg[i]->SpecialInfo  =
  (APTR)&gd->PrInfo[i-22];
  gd->Gadg[i]->GadgetRender =
  (APTR)&gd->dummyImg[i-22];
 }
 InitGadget(22,237,30,17,26,gd);
 InitGadget(23,237,86,17,53,gd);
 for(i = 0; dl[i] != NULL && i < MAXDEVS; i++) {
  InitGadget((WORD)(i+24),10,
  (WORD)(19 + i * 24),68,24,gd);
  gd->Gadg[i+24]->GadgetType   |= BOOLGADGET;
  gd->Gadg[i+24]->Flags        |= GADGIMAGE|GADGHIMAGE;
  gd->Gadg[i+24]->GadgetText   = gd->GadgText[i+16];
  gd->Gadg[i+24]->GadgetRender = (APTR)&gd->FloppyImg;
  gd->Gadg[i+24]->SelectRender =
  (APTR)&gd->SelFloppyImg;
  gd->GadgText[i+16]->IText    = dl[i];
  gd->GadgText[i+16]->FrontPen = 3;
  CenterIText(gd->GadgText[i+16],2,6,68,24);
  last = gd->Gadg[i+24];
 }
 last->NextGadget = NULL;
 InitBorder(gd->Rvec,0,0,REQWIDTH - 2,REQHEIGHT - 1);
 InitBorder(gd->Dvec,86,19,252, 66);
 InitBorder(gd->Fvec,86,75,252,149);
 gd->headertxt[0].IText =
 gd->headertxt[1].IText = text;
 CenterIText(&gd->headertxt[0],0,1,REQWIDTH,18);
 gd->headertxt[1].LeftEdge =
 gd->headertxt[0].LeftEdge - 2;
 gd->headertxt[1].TopEdge =
 gd->headertxt[0].TopEdge  + 1;
 gd->filereq.LeftEdge =
 MAX((gd->dispwindow->Width - REQWIDTH)  >> 1,0);
 gd->filereq.TopEdge =
 MAX((gd->dispwindow->Height - REQHEIGHT) >> 1,0);
 gd->filereq.Width      = REQWIDTH;
 gd->filereq.Height     = REQHEIGHT;
 gd->filereq.ReqGadget  = gd->Gadg[0];
 gd->filereq.BackFill   = 2;
 gd->filereq.ReqBorder  = &gd->FileBorder;
 gd->filereq.ReqText    = &gd->headertxt[1];
 return(gd->Gadg[0]);
}

extern UBYTE* __saveds BuildFileRequester
(UBYTE *dirname,UBYTE *nodename,UBYTE *filename,
struct Window *w,UBYTE *title) {
 struct IntuiMessage *imsg;
 struct Buffer *buffer;
 struct Process *proc;
 struct DATA *gd;
 struct MsgPort *OldPort = w->UserPort;
 ULONG sm,class,OldIDCMPFlags = w->IDCMPFlags;
 APTR oldwindow;
 BPTR cdlock,plock;
 UWORD gadg;
 UBYTE *dname,*string;
 WORD i;
 if(w == NULL || dirname == NULL || nodename == NULL
 || filename == NULL) return((UBYTE*)-1);
 if(gd = InitGlobal()) {
  gd->dispwindow = w;
  gd->StrInfo[0].Buffer = gd->dirbuf = dirname;
  gd->StrInfo[1].Buffer = gd->filebuf = nodename;
  gd->StrInfo[0].UndoBuffer =
  gd->StrInfo[1].UndoBuffer = gd->undobuf = filename;
  gd->StrInfo[0].MaxChars = STRSIZE;
  gd->StrInfo[1].MaxChars = MAXNAME;
  proc = (struct Process*)FindTask(NULL);
  oldwindow = proc->pr_WindowPtr;
  proc->pr_WindowPtr = (APTR)gd->dispwindow;
  FindDevices(gd);
  if(AllocGadgets(gd->namelist,
  w->WScreen->Font->ta_YSize,title,gd)) {
   if(gd->dirbuf[0] == NULL)
    BldPath(Lock(":",ACCESS_READ),gd->dirbuf);
   w->UserPort = NULL;
   ModifyIDCMP(w,GADGETUP|GADGETDOWN|REQSET);
   sm = 1L << w->UserPort->mp_SigBit;
   if(Request(&gd->filereq,w) == TRUE) {
    FOREVER {
     WaitPort(w->UserPort);
     if(imsg = (struct IntuiMessage*)
     GetMsg(w->UserPort)) {
      class  = imsg->Class;
      gadg   = ((struct Gadget*)
      imsg->IAddress)->GadgetID;
      ReplyMsg((struct Message*)imsg);
```

Listing: Requester-Library

```
      if(class == REQSET) {
       ScrollField(NULL,ET_DIR,gd);
       ScrollField(NULL,ET_FILE,gd);
       ActivateGadget(gd->Gadg[21],w,&gd->filereq);
       buffer = GetDir(gd->dirbuf,sm,gd);
      }
      else if(class == GADGETUP) {
       dname  = ((struct Gadget*)
       imsg->IAddress)->GadgetText->IText;
       string = (UBYTE*)gd->Gadg[gadg]->UserData;
       if(gadg == 0 || gadg == 21) {
        strcpy(gd->undobuf,gd->dirbuf);
        if((i = strlen(gd->undobuf)) > 0
        && gd->undobuf[i - 1] != ':')
         strcat(gd->undobuf,"/");
        strcat(gd->undobuf,gd->filebuf);
        break;
       } else if(gadg > 23) {
        BldPath(Lock(dname,ACCESS_READ),gd->dirbuf);
        RemakeGadgs(gd->Gadg[20],gd->Gadg[20],gd);
        buffer = GetDir(gd->dirbuf,sm,gd);
       } else if(gadg == 20) {
        buffer = GetDir(gd->dirbuf,sm,gd);
       } else if(gadg == 2) {
        filename = NULL; break;
       } else if(gadg == 1) {
        if(cdlock = Lock(gd->dirbuf,ACCESS_READ)) {
         if(plock = ParentDir(cdlock)) {
          BldPath(plock,gd->dirbuf);
          RemakeGadgs(gd->Gadg[20],gd->Gadg[20],gd);
          buffer = GetDir(gd->dirbuf,sm,gd);
         }
         UnLock(cdlock);
        }
       } else if(gadg > 2 && gadg < 8 && string) {
        if(gd->dirbuf[strlen(gd->dirbuf) - 1] != ':')
         strcat(gd->dirbuf,"/");
        strcat(gd->dirbuf,string);
        BldPath(Lock(gd->dirbuf,
        ACCESS_READ),gd->dirbuf);
        RemakeGadgs(gd->Gadg[20],gd->Gadg[20],gd);
        buffer = GetDir(gd->dirbuf,sm,gd);
       } else if(gadg > 7 && gadg < 16 && string) {
        strcpy(gd->filebuf,string);
        RemakeGadgs(gd->Gadg[21],gd->Gadg[21],gd);
       } else if(gadg == 22) {
        gd->doff = ReadProp(gd->dcount,22,MAXDIR,gd);
        gd->dcount = ScrollField(buffer,ET_DIR,gd);
       } else if(gadg == 23) {
        gd->foff = ReadProp(gd->fcount,23,MAXFILE,gd);
        gd->fcount = ScrollField(buffer,ET_FILE,gd);
       }
      } else if(class == GADGETDOWN) {
       do {
        switch(gadg) {
         case 18: if(gd->dcount - MAXDIR -
          gd->doff <= 0) break;
          gd->doff += 2;
         case 19: if(gd->doff == 0) break;
          gd->doff--;
          gd->dcount = ScrollField(buffer,ET_DIR,gd);
          break;
         case 16: if(gd->fcount - MAXFILE -
          gd->foff <= 0) break;
          gd->foff += 2;
         case 17: if(gd->foff == 0) break;
          gd->foff--;
          gd->fcount = ScrollField(buffer,ET_FILE,gd);
          break;
         default: break;
        }
       } while(gd->Gadg[gadg]->Flags & SELECTED);
      }
     }
    }
   } else filename = (UBYTE*)-1;
   ModifyIDCMP(w,NULL);
   w->UserPort = OldPort;
   ModifyIDCMP(w,OldIDCMPFlags);
  } else filename = (UBYTE*)-1;
  FreeRemember(&gd->Rkey,TRUE);
  proc->pr_WindowPtr = oldwindow;
  FreeMem((UBYTE*)gd,sizeof(struct DATA));
 } else filename = (UBYTE*)-1;
 return(filename);
}
```




```
/*===================================================*
*==                                                ==*
*==       FileRequesterImages.C                    ==*
*==                                                ==*
*==       by Franz-Josef Reichert                  ==*
*==                                                ==*
*==       Lattice C-Compiler V 5.04                ==*
*==                                                ==*
*===================================================*/

#include <exec/types.h>

UWORD __chip floppy[] = {
 0x0fff,0xffff,0xffff,0xfffe,0x0000,0x7fff,0xffff,
 0xffff,0xffff,0xc000,0xffff,0xc000,0x0000,0x3fff,
 0xe000,0xff00,0x0000,0x0000,0x000f,0xe000,0xffff,
 0xc000,0x0000,0x3fff,0xe000,0xffff,0xffff,0xffff,
```

Listing: Requester-Library

```
 0xffff,0xe000,0xffff,0xffff,0xffff,0xeaaf,0xe000,
 0xffff,0xffff,0xffff,0xeaaf,0xe000,0xffff,0xffff,
 0xffff,0xffff,0xe000,0x57fd,0x5555,0x5555,0x5555,
 0x4000,0x7fff,0xffff,0xffff,0xffff,0xc000,0x0fff,
 0xffff,0xffff,0xfffe,0x0000,0xf000,0x0000,0x0000,
 0x0001,0xe000,0x8000,0x0000,0x0000,0x0000,0x2000,
 0x0000,0x3fff,0xffff,0xc000,0x0000,0x0055,0x5555,
 0x5555,0x5550,0x0000,0x0000,0x0000,0x0000,0x0000,
 0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,
 0x0000,0x0000,0x1110,0x0000,0x0000,0x0000,0x0000,
 0x0440,0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,
 0x83f8,0x0000,0x0000,0x0000,0x2000,0x8000,0x0000,
 0x0000,0x0000,0x2000,0xf000,0x0000,0x0000,0x0001,
 0xe000
};
UWORD __chip selfloppy[] = {
 0x0000,0x0000,0x0000,0x0000,0x0000,0x03ff,0xffff,
 0xffff,0xfff8,0x0000,0x1fff,0xffff,0xffff,0xffff,
 0x0000,0x3fff,0xf000,0x0000,0xffff,0x8000,0x3fc0,
 0x0000,0x0000,0x003f,0x8000,0x3fff,0xf000,0x0000,
 0xffff,0x8000,0x3fff,0xffff,0xffff,0xffff,0x8000,
 0x3fff,0xffff,0xffff,0xffff,0x8000,0x15ff,0x5555,
 0x5555,0x5555,0x0000,0x1fff,0xffff,0xffff,0xffff,
 0x0000,0x03ff,0xffff,0xffff,0xfff8,0x0000,0x0000,
 0x0000,0x0000,0x0000,0x0000,0xf000,0x0000,0x0000,
 0x0001,0xffff,0x8000,0x0000,0x0000,0x0000,0x3fff,
 0x0000,0x0000,0x0000,0x0000,0x1fff,0x0000,0x0fff,
 0xffff,0x0000,0x1fff,0x0015,0x5555,0x5555,0x5540,
 0x1fff,0x0000,0x0000,0x0000,0x0000,0x1fff,0x0000,
 0x0000,0x0000,0x0000,0x1fff,0x0000,0x0000,0x0000,
 0x0000,0x1fff,0x00fe,0x0000,0x0000,0x0000,0x1fff,
 0x0000,0x0000,0x0000,0x0000,0x1fff,0x8000,0x0000,
 0x0000,0x0000,0x3fff,0xf000,0x0000,0x0000,0x0001,
 0xffff
};
UWORD __chip arrowup[] = {
 0xffff,0x8000,0xc001,0x8000,0xc081,0x8000,0xc1c1,
 0x8000,0xc3e1,0x8000,0xc7f1,0x8000,0xcff9,0x8000,
 0xc3e1,0x8000,0xc3e1,0x8000,0xc3e1,0x8000,0xc001,
 0x8000
};
UWORD __chip arrowdown[] = {
 0xc001,0x8000,0xc3e1,0x8000,0xc3e1,0x8000,0xc3e1,
 0x8000,0xcff9,0x8000,0xc7f1,0x8000,0xc3e1,0x8000,
 0xc1c1,0x8000,0xc081,0x8000,0xc001,0x8000,0xffff,
 0x8000
};
```

```
/*===================================================*
*==                                               ==*
*==       Global.C                                ==*
*==                                               ==*
*==       by Franz-Josef Reichert                 ==*
*==                                               ==*
*==       Lattice C-Compiler V 5.04               ==*
*==                                               ==*
*===================================================*/

#include <intuition/intuition.h>
#include <proto/intuition.h>

struct TextAttr taPlain = {"topaz.font",TOPAZ_EIGHTY};

VOID InitBorder(WORD *p,WORD x,
WORD y,WORD width,WORD height) {
 p[0]  = p[2]  = x;
 p[8]  = p[10] = x + 1;
 p[1]  = p[7]  = p[9]  = p[15] = y;
 p[3]  = p[5]  = p[11] = p[13] = height;
 p[12] = p[14] = width;
 p[4]  = p[6]  = width + 1;
 return;
}

VOID CenterIText(struct IntuiText *it,
WORD absleft,WORD abstop,WORD w,WORD h) {
 it->LeftEdge = absleft +
 ((w - IntuiTextLength(it)) >> 1);
 it->TopEdge = abstop +
 ((h - it->ITextFont->ta_YSize) >> 1);
 return;
}
```




```
/*===================================================*
*==                                               ==*
*==       Requester.H                             ==*
*==                                               ==*
*==       by Franz-Josef Reichert                 ==*
*==                                               ==*
*==       Lattice C-Compiler V 5.04               ==*
*==                                               ==*
*===================================================*/

#ifndef REQUESTER_H
#define REQUESTER_H

#include "exec/types.h"
#include "libraries/dos.h"
#include "exec/libraries.h"
#include "intuition/intuitionbase.h"
#include "graphics/gfxbase.h"
```

Listing: Requester-Library

```

#define REQUESTERNAME "requester.library"

struct RequesterBase {
 struct Library        rb_LibNode;
 BPTR                  rb_SegList;
 struct ExecBase       *rb_SysBase;
 struct IntuitionBase  *rb_IntuitionBase;
 struct GfxBase        *rb_GfxBase;
 struct DOSBase        *rb_DOSBase;
};

extern struct RequesterBase *RequesterBase;

#ifndef __ARGS
#ifdef NARGS
#define __ARGS(a) ()
#else
#define __ARGS(a) a
#endif NARGS
#endif !__ARGS

#ifndef LATTICE_50
#define __saveds
#define __regargs
#endif !LATTICE_50

extern UBYTE* __saveds BuildFileRequester
__ARGS((UBYTE*,UBYTE*,UBYTE*,struct Window*,UBYTE*));
extern UWORD* __saveds BuildColorPalette
__ARGS((UWORD*,struct Screen*,UBYTE*));
extern LONG __saveds BuildBoolRequester
__ARGS((LONG,LONG,LONG,LONG,struct TextAttr*,
struct Window*,UBYTE**,UBYTE**));

#ifndef AZTEC_C
#ifndef NO_PRAGMAS
#pragma libcall RequesterBase BuildFileRequester 1e 982
1005
#pragma libcall RequesterBase BuildColorPalette 24 9800
3
/*pragma libcall RequesterBase BuildBoolRequester 2a 98
321008*/
#endif !NO_PRAGMAS
#endif !AZTEC_C

#endif !REQUESTER_H
```

```
/*===================================================*
*==                                               ==*
*==       Minex.C                                 ==*
*==                                               ==*
*==       by Franz-Josef Reichert                 ==*
*==                                               ==*
*==       Lattice C-Compiler V 5.04               ==*
*==                                               ==*
*===================================================*/

#include <exec/types.h>
#include <intuition/intuition.h>
#include <proto/exec.h>
#include <proto/intuition.h>
#define NO_PRAGMAS
#include "requester.h"

extern struct IntuitionBase *IntuitionBase;
static UBYTE pathname[128],nodename[32],filename[256];
struct NewWindow nw = {
 0,0,640,256,0,1,CLOSEWINDOW|MOUSEBUTTONS,
 WINDOWDRAG|WINDOWCLOSE|WINDOWDEPTH|SMART_REFRESH|
 NOCAREREFRESH|WINDOWSIZING|GIMMEZEROZERO,NULL,NULL,
 "TestWindow",NULL,NULL,100,50,640,512,WBENCHSCREEN
};

main(int argc,UBYTE *argv[]) {
 struct IntuiMessage *msg;
 struct Window *w;
 ULONG class = 0; UWORD code;
 UBYTE *name;
 if(IntuitionBase = (struct IntuitionBase*)
 OpenLibrary("intuition.library",LIBRARY_VERSION)) {
  if(w = (struct Window*)OpenWindow(&nw)) {
   if(argc > 1) strcpy(pathname,argv[1]);
   if(argc > 2) strcpy(nodename,argv[2]);
   while(class != CLOSEWINDOW) {
    WaitPort(w->UserPort);
    while(msg = (struct IntuiMessage*)
    GetMsg(w->UserPort)) {
     class = msg->Class;
     code  = msg->Code;
     ReplyMsg(&msg->ExecMessage);
     if(class == MOUSEBUTTONS && code == SELECTUP) {
      if((name = BuildFileRequester(pathname,nodename,
  filename,w,"FJR's File Requester Deluxe")) > 0)
       printf("%s\n",name);
     }
    }
   }
   CloseWindow(w);
  }
  CloseLibrary((struct Library*)IntuitionBase);
 }
}
```

Listing: Requester-Library

GNE

5.25" TEAC Profilaufwerk extern 279,-
1.67 MB unformatiert, 880 KB im AMIGAFORMAT, 100% kompatibel zu internem 3.5", Diskchange, abschaltbar, 40/80 Track unschaltbar original standartdmäßiger durchgeschliffener Bus, Amigafarbenes extrem robustes Stahlblechgehäuse, direkt MS-DOS + PC/XT/AT Karten kompatibel, mindestens 80 cm langes zugentlastetes Rundkabel, Stromversorgung über Amiga, passender BOOTSELEKTOR im Preis von 279,- enthalten, inklusive WRITE PROTECT Schalter, qualitätsgeprüft, ausführliche deutsche Bedienungsanleitung

5.25" TEAC intern A2000 239,-
Technische Daten wie 5.25" extern, wahlweise DF1 oder DF2. DF2 kostet 259,- wegen zusätzlicher Modifikationsplatine! unkomplizierter Einbau innerh. 2 Minuten OHNE LÖTEN!!!

3.5" TEAC Profilaufwerk extern 229,-
Technische Daten wie 5,25" Laufwerk extern, Write Protect Schalter und Bootselektor im Preis enthalten!

3.5" TEAC intern A2000 169,-
Technische Daten kompatibel zu 3.5" Laufwerk extern! für A500 als DFO, für A2000 als DFO-DF1, für A1000 als DFO! unkomplizierter Einbau innerhalb 2 Minuten OHNE LÖTEN!!!

BOOTSELEKTOR DF1/DF2/DF3 15,-
SOUNDVERTEILER 19,-
A500 512KB + Uhr/abschaltbar 179,-
STAR LC 24-10 Farbband STAR LC 24-10: 9,- DM 698,-
STAR XB 24-10 1398,-
NEC P2 PLUS Farbband NEC P2200: 8,50 DM 849,-
NEC P6 PLUS Farbband NEC P6/P6+: 9,- DM 1349,-
Umschaltplatine mit Kick 1.3 98,-

Wir liefern nur deutsche Geräte mit Seriennummer und Super-Service!
1 Jahr Garantie auf alle Produkte dieser Anzeige!

GNE - GREBE NEUMANN ELEKTRONIK
Am Stein 10, 5419 Raubach, 02684-5566/5572
Telex: 869987 **Fax: 02684/5448**
HÄNDLERANFRAGEN ERWÜNSCHT!!!



Wie?... Wir verschenken keine Amigas! Aber wir haben immer gute Angebote! Von A 500 bis A 2500, ob Filecard ... oder PC-AT-Karten oder Zubehör

Was?... Alles für den Amiga bis Professional! Anrufen lohnt sich!!!

Wo?... P.V. Computer-Shop
Bahnhofstr. 278, 8623 Wetzikon (ZH)
Tel. 01/9 30 79 54
Schweiz

HARDWARE • SOFTWARE • BERATUNG

Commodore® Ersatzteil Service

✖ Wir liefern für **Händler** und Privatanwender preiswert und prompt

✖ Rufen Sie uns an: (02331-43001)
oder schreiben Sie uns:

CIK-Computertechnik • Ingo Klepsch
Berliner Straße 49b • D-5800 Hagen 7
TELEFAX: 02331-42499

ALSDORFER PUBLIC DOMAIN CENTER

Wir liefern u. a. die Serien:
ACS, Amicus, AmSel, Auge, Bavarian, Cactus, Faug, Fish, Franz, Kickstart, Panorama, RPD, Ruhr, Saar, Taifun u.v.a.

Unsere Preise inkl. Etiketten

3,5'' 2DD No Name	5,25'' 2D No Name
1– 50 à **2,50 DM**	1– 50 à **1,50 DM**
51–100 à **2,40 DM**	51–100 à **1,40 DM**
ab 101 à **2,30 DM**	ab 101 à **1,30 DM**

Leerdisketten inkl. Etiketten

3,5'' 2DD No Name	5,25'' 2D No Name
10– 50 à 10 St. **15,50 DM**	10– 50 à 10 St. **8,50 DM**
51–100 à 10 St. **14,50 DM**	51–100 à 10 St. **7,50 DM**
ab 101 à 10 St. **13,50 DM**	ab 101 à 10 St. **6,50 DM**

Innerhalb **48 Std.** verläßt Ihre Bestellung unser Haus.
Versandkosten 8 DM + NN, Ausland nur Vorkasse + 15 DM

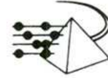
WALTER KAMINSKI
Geilenkirchener Straße 4 - 5110 Alsdorf
Telefon 02404/22963

Tino Hofstede
Computerservice
An der Windmühle 8
5010 Bergheim 5

WARRIORS, SF-Adventure	19,90
HOTELMASTER, Spiel	19,90
ENGLISCHTRAINER incl. Vokabeln	39,90
FRANZÖSICHTRAINER incl. Vokabeln	39,90
VIDEOVERWALTUNG	49,90
DISK-ETIKETTEN-Druck/Programm	19,90
LIGAVERWALTUNG (6-20 Mannschaften)	49,90
PD-DISKETTEN	ab 2,60
KATALOG (Schutzgebühr in Briefmarken)	1,00

Daten- und Organisationssysteme
Hard- und Softwarevertrieb

Ihr AMIGA-Fachhändler im Bergischen Land!



Filecard A-2000 AutoBoot
21 MB 40ms MFM............999 DM
32 MB 40ms RLL............1.099 DM
47 MB 40ms RLL............1.459 DM
47 MB 28 ms RLL...........1.499 DM
88 MB 15 ms MFM........2.849 DM
133 MB 15ms RLL..........3.399 DM
Alle Cards inkl. ALF 2.0 Software, AutoPark, AutoBoot unter Kick 1.3, Einbau im Preis.

HardDisk A-500/1000 AutoBoot
20 MB 40ms MFM..............1.149 DM
32 MB 40ms RLL.................1.239 DM
43 MB 28ms MFM..............1.599 DM
48 MB 28ms RLL.................1.649 DM
65 MB 28ms RLL.................1.799 DM
Alle Disks inkl. ALF 2.0 Software, Gehäuse, Netzteil
Amiga Expansionsport ist durchgeführt, Anschlußkabel

Preisgünstige Speichererweiterungen
abakus 512KB, m. Akku, abschaltbar........199 DM
A 508 2MB-Card, Akku, abschaltb., 512KB bestückt........349 DM
A 508 2MB-Card, Akku, abschaltb., 2MB bestückt........649 DM
2-MB-Box, Akku, ab- u. umschaltb.,512KB bestückt........449 DM
4MB-Box, Akku, ab- u. umschaltb., 2 MB bestückt........999 DM
MICROBOTICS 8-UP 2,4,6,8MB Betrieb, 2 MB bestückt........899 DM
MICROBOTICS 8-UP 8 MB bestückt nur........1.999 DM

Kickstart-Umschaltplatinen
Kick-Umschaltplatine ROM/ROM mit ROM 1.2........69,90 DM
Kick-Umschaltplatine ROM/ROM mit ROM 1.3........99,90 DM
Kick-Umschaltplatine ROM/EPROM........69,90 DM
BootStrap, mit RAM, mit WOM, mit Gary Adap........339 DM

Öffnungszeiten (Büro + Ladengeschäft)
Mo – Fr 10.00 – 18.30 – Sa 10.00 – 14.00 – langer Sa 10.00 – 16.00
Sedanstraße 136 * 5600 Wuppertal 2
Tel. 0202/ 501500 * Martin Kramer

SUPERPACK 50

PD der Extraklasse! 50 (!)
ausgesuchte Programme [redacted] Return to Earth, Kampf um Eriador, Risk, Broker, Paranoid, Lucky Loser, Faktura, MS-Text, Videodatei, Plattenliste, Superliga, Haushaltsbuch, MCAD, Wizard of Sound, CLI-Pack, Virus Stop, Werner-Spiel, Latein, R.O.M., Star Trek, Alf, Core Wars, Label, Amiga-Paint, Giroman. Blizzard, VirusControl, Tetrix, Moria, Battleforce, Peters Quest, Super-Bilder, Billard, Einkommensteuer, DSort III, Fix-Disk, Universaldatei, Quickmenü, Diskey, Mandelbrot, Silver Bilder, Astron.,Super-Print,Calc,Atlantis,Schach,Labelpaint!

Alle Programme zusammen kosten DM **69,-**

Alle gängigen PD-Serien lieferbar!

Katalogdisk **GRATIS!**

Pawlowski-Software-Service
Ellerbruch 19, 2177 Wingst, 04778/7294
Versandkosten: Vorkasse 3,50 DM, NN 6,00 DM

DATATRANS

Endlich die Daten Übertragungsmöglichkeit:

Vom Commodore C64 / C128 zum AMIGA 500/1000/2000

Das Kit besteht aus folgenden Komponenten:
1.) Software Diskette für C 64 / C 128
2.) Übertragungskabel Länge ca. 2 Meter
3.) Anpassungsprogramm für die Empfangsseite
4.) Ausführliche Bedienungsanleitung
5.) Tel. Hotline für spezielle Fragen



Unser Übertragungskit ist auch für PC's + ATARI-ST's bestimmt!

Preis 60,-- DM incl. Versandkosten.

Händler Anfragen erwünscht!

HVK Kunze Hollandstr. 119 4600 Dortmund 12
Tel. 0231/259090
FAX0231/201565

(c) by Welte Soft

VIRUS DETEKTOR PROFESSIONAL

Balsam für die Nerven

Erhältlich im Fachhandel - Bezugsquellennachweis bei

DIT

Dienstleistungen und Informationstechnologie
Bert Schwarm
Musfeldstraße 1
4200 Oberhausen 11
Telefon : 0208 605645

AMIGA DOS -TIP – die aktuelle Produktanzeige

VEREIN — AMIGA

Programmsystem, bestehend aus 7 Einzelprogrammen, nämlich: Editor für Grunddateneingabe, Mitgliederdatei mit Beitragsübersicht, Adreßetiketten- + Rundschreiben-Eindruck, 80-Zeichen-Liste u. codierte Auszüge – Mahnung, Lieferanten Bestellung – Datei der befreundeten Vereine und Turniergegner mit Listen, Etiketten usw. wie vor – Turnierverwaltung mit Score-Saldo + Terminverwaltung – Inventar/Inventur des Vereinsvermögens – Vereinskasse mit Belegausdruck und Kassenprotokoll auf Disk/Drucker – Kassenstand-Blitzanzeige – Einnahme+Ausgabe-- Überschuß-Rechnung – Bildschirm-Rechner

Speicherbedarf 512 kByte – Keine besonderen Druckeranforderungen – Übersichtlich, schnell, auch lieferbar für ATARI ST, PC/XT/AT

Preis N U R DM 198.-

Versandkosten pro Sendung : Nachnahme DM 5.70, Ausland DM 10.70; Vorkasse DM 3.-
Liste gegen adressierten Freiumschlag DINA5/DM1.-
Händler sehr erwünscht.



I. DINKLER
Am Schneiderhaus 7
Tel. 02932/32947
D-5760 ARNSBERG 1

DONAU-SOFT
24 h-Schnellversand

Neutrale Disketten
3,5" 2DD (100 % errorfree)

	von Sentinel	von SONY/Collossus
bis 99 Stück	1,60 DM	2,00 DM
ab 100 Stück	1,40 DM	1,85 DM
ab 500 Stück	**1,25 DM**	**1,70 DM**

Laufwerke mit allen Extras
3,5" intern .... 155,– DM
3,5" extern, abschaltbar, Busdurchführung .... 209,– DM
5,25" extern, wie 3,5" + 40/80-Trackumschaltung .... 269,– DM

Sim City (dt.)	79,– DM	GFA-Basic 3.5	208,– DM
B.A.D.	77,– DM	GFA-Compiler 3.5	129,– DM
DPaint III dt.	240,– DM	512 KB-Erw. (A500)	199,– DM
Zoetrope 1.1	189,– DM	2 MB-Erw. (A500)	598,– DM

Vorkasse: + 5,– DM, Nachnahme: + 8,– DM, Ausland: + 10,– DM

MAIK HAUER
Postfach 1401, 8858 Neuburg Fax: 08431/49800
Tel.: 08431/49798 (bis 22 Uhr) BTX: *Donau-Soft #

AMIGA - PD

Wir liefern nur ORIGINALE AMIGA PUBLIC DOMAIN!!!
aus einem Bestand von über

! 8000 DISKETTEN !
Kopiert auf 3,5"....... ab 2,60
Kopiert auf 5,25"....... ab 1,40

5 deutsche Katalog-Disk. (3,5") + die neueste TIME: DM 20,– (VK)

DIVERSE SONDER-PD! z.B.:

27 Disk.	Soundtracker-Modules	St. 4,–
42 Disk.	AGATron (T. Richter)	St. 4,–
9 Disk.	Olli's Games	St. 4,–
98 Disk.	UGA (holl. TOP-Serie)	St. 4,50
224 Disk.	SEKA-Sources	St. 4,–
15 Disk.	Share! (Shareware)	St. 5,–

A.P.S. -electronic-
Sonnenborstel 31
D-3071 Steimbke
Tel.: 05026/1700
FAX 05026/1615



INFO-DISKETTE 2,– (Briefmarken)

AMIGA — Astrol. Kosmogramm

Nach Eingabe von Namen, Geburtsort (geogr. Lage) + -datum werden errechnet: Sternzeit, Aszendent+ Medium Coeli, Zodiakradianten, 12 Objektpositionen im Tierkreis, Koch/Schaeck-Häuser, Aspekte * Allgem. Persönlichkeitsanalyse mit Ideal-Partner Skala, Bild-/Druckerausgabe 3 DINA4-Seiten, Horoskop-Diagramm – Alle Planeten +Sonne+Mond, Mondknoten – Minutengenaue Berechnung – Sommerzeiten + Koordinaten-Einlesung DM 78.-

BIOKURVEN — AMIGA

Wissenschaftl. Trendbestimmung der biologischen+ seelisch/geistig/körperlichen Rhytmik – Monitor-Ausgabe monatsweise vor- +rückschreitend, Ausgabe Drucker beliebig lang mit täglischer Analyse und Kennzeichnung kritischer Tage – Absolut-+Mittelwerte – Ideal für Partnervergleich – Text-Editor für Zusätze – Wissensch. Grundlagen DM 58.-

AMIGA — Kalorien-Polizei

Erstellung von Diätplänen und personbezogene Bedarfsrechnung auf Eingabe von Größe, Gewicht, Geschlecht, Leistung – Energiebilanz nach Fett, Eiweiß, Kohlenhydraten – Ideal-/Über-/Untergewicht Einlesung Vitalstoffe, Kalorien-Lebensmittel-Tabelle, Aktivitäten+Verbrauch – Bildschirm- +oder Druckerausgabe auf einigen DINA4 DM 58.-

Etikettendruck — AMIGA

Druckt 40 gängige Haftetiketten-Formate nach Gestaltung in jeweils passender Bildschirm-Maske + Bestimmung der Auflagehöhe – Ablage auf Disk für sofortige Neu-Auflage – Schriftenwahl DM 88.-

****
IDEE-SOFT-Programme
– Exzellent in ihrer Struktur – alle Programme in Deutsch –

AMIGA — Registrierkasse

+Normaldrucker. Beleg auf Tab.Papier 145mm- Kassenführung auf Disk für Ausdruck+Unterbrechung – Artikel/Dienstl. von Disk abrufbar – Einbindung von Firmendaten, Werbeslogans o.ä. – m/o MWSt. – Ideal für alle Gewerbe mit Bareinnahmen DM 148.-

GESCHÄFT — AMIGA

Editor für Formular-, Adressen-, Artikel-Dienstleistungsdateien – Optionen: Angebot/Kosten-Voranschlag, Auftrag/Bestellung, Auftr.Bestatigung, Rechnung, Lieferschein, Mahnung. Eingabe Hand o. Datei – 20 Positionen/DINA4 Durchrechn.über Menge, Preis, Aufschlag/Rabatt, MWSteuer, Skonto -- Texteditor für Zusätze – Kein Verbund zu Lager-/FIBU – Schnell, übersichtlich, Userfrdl. DM 198.-

AMIGA — Inventur, Fibu-gerecht

Kontinuierl. Bestandsverwaltung m.Bildmoment u/o Listenauswertung – Neu-Inventur durch Streichen, Ändern, Hinzufügen – Gruppeninventur nach Code – 1000 Positionen/Liste – Blätteraddition DM 118.-

Provisionsabrechnung — AMIGA

Editor für Vertreter, Kunden-, Formulardaten- 25 Positionen/DINA4, Eingabe Hand/Datei -PSatz 0.01 -99.99% – Storno, Spesengutschr. – Durchrechnung zum Endbetrag, m/o MWSteuer – schnell ! DM 118.-

AMIGA — TYPIST

AMIGA als elektronische Schreibmaschine mit zeilenweisem Ausdruck und 15zeiligem Bildschirmdisplay – Je nach Druck bis zu 30 Schriftarten -File auf Disk – Kopie-Ausdruck – Super ! DM 88.-

****
IDEE-SOFT-Programme
– Exzellent in ihrer Struktur – alle Programme in Deutsch –

AMIGA — GELD

30 Routinen für Umgang mit Geld: Anlage – Vermögensbildung – Rentensparen – Rendite – Kredite – Lasten – Zinsen – Hypothek – Laufzeit – Amortisation – Raten – Gleitklausel – Nominal/Effektiv Zins – Akonto+Restverzinsung – Diskont – Konvertierung – kpl.Tilgungspläne Bild/Druck DM 98.-

DATEIVERWALTUNG — AMIGA

Datenfelder von je 8 Zeilen a 33 Zeichen, je Datei max.1000 – Suchcode von max.33 Zeichen, mit jedem mehr die Zielgruppe einengend – Optionen : Code, Nummer, alle, Blatt vor/zurück, Streichen, Ändern(zeilenweise), Hinzufügen – Druck: 80-Zeichen-/Blockliste, Seitenvorschub, Etiketten, Datenfeld-Maske – Gezielte Aufgaben, superschnell-Übersichtlich, bedienerfreundlich, mausgesteuert

Adressen	68.-	**Galerie**	118.-
Bibliothek	118.-	**Lager**	118.-
Briefmarken	118.-	**Personal**	118.-
Diskothek	78.-	**Stammbaum**	118.-
Exponate	118.-	**Videothek**	78.-

DEFIN DATA zum Selbstdefinieren der Inhalte DM 148.-

****
Versandkosten pro Sendung : Nachnahme DM 5.70, Ausland DM 10.70; Vorkasse DM 3.-
Liste gegen adressierten Freiumschlag DINA5/DM1.-
Händler sehr erwünscht.



I. DINKLER
Am Schneiderhaus 7
Tel. 02932/32947
D-5760 ARNSBERG 1

Speichererweiterungen

ANRUFEN !!!! LOHNT SICH !!!!
TAGESPREISE !!!!!

Amiga 500	- 512 KB intern, abschaltbar	ab	DM	159,--
Amiga 500	- 1 MB intern, absch , Uhr	ab	DM	398,--
Amiga 500	- 2 MB intern, absch , Uhr	ab	DM	598,--
Amiga 1000	- 2 MB extern, absch,	ab	DM	798,--
Amiga 2000	- 2 MB intern, absch,	ab	DM	898,--

Alle Speichererweiterungen sind autokonfigurierend, abschaltbar und mit sehr schnellen RAM's (100ns und schneller) ausgerüstet!
Durch Megabit-Technologie minimaler Strombedarf.
****12 Monate Garantie****

Laufwerk 3.5" intern f. Amiga 2000 DM 169,--
Laufwerk 3.5" extern, durchgeschl. Bus, abschaltb. DM 198,--

B & S Computer-Vertriebs GmbH
Beethovenstr. 33 ; 4172 Straelen 1
Tel: 02834/1249 ; Fax: 02834/6979

MacSoft – *AMIGA SHOP*
Hardware – Software – Schulung – PD

Public Domain
DISKETTE AUF 2 DD NUR 4,– DM

Über 4500 PD-Disk! Immer aktuell!
24-Std.-Versand-Service
Katalog-Disketten anfordern, 5,– DM
Selber abholen, NN gespart!
Hardware-Zusammenstellung auf Wunsch.
Lassen Sie sich Ihren persönlichen Amiga anfertigen.
Fragen Sie nach unseren Amiga-Einsteiger-Kursen.

Telefon 0231/516010
Mo.-Fr. 10–13, 15–20 Uhr, Sa. 10–16 Uhr
Hannövrischestr. 82, 4600 Dortmund 1
*BTX * mac soft amiga #*

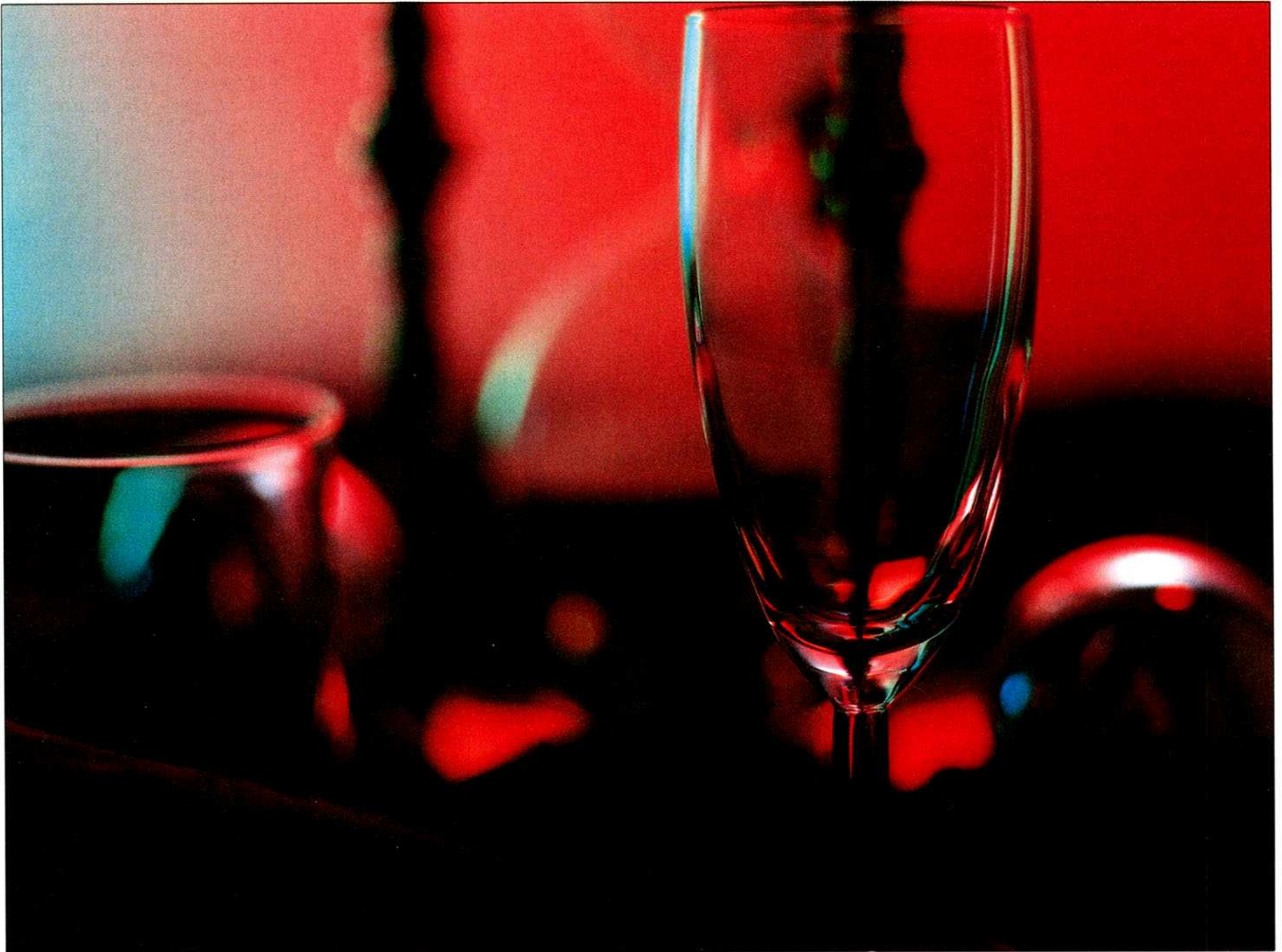
5000
AMIGA Public-Domain
80 SUPER AMIGA-PD Serien
Ultraschnell-Aktuell-Preiswert-Zuverlässig
Fish, Kickstart, TBAG, Franz, Taifun Antares...

Seit 1985 haben wir AMIGA-PD

Incl. 3,5 MF2DD Disk
ab 1 - 79 Disk a 2,00 DM
3,5 MF2DD schon ab 1,60 DM
Incl. 5,25 Disk
10 - 99 a 1,20, ab 100 a 1,00 DM

Wir kopieren mit doppeltem Verify.
AMIGA PD incl. Markendisk auf Anfrage.
Katalogdiskette 3,- DM, bei Vorkasse in Briefm.
oder fordern Sie unser Info-Material an 1,- DM
Montag-Freitag 10.00-20.00, Samstag 11.00 - 16.00
Tel: 040/6428225, Telefax: 040/6426913
Vork. + 4,- DM, Nachn. + 8,- DM, Ausland + 16,- DM

R. Dombrowski
Postfach 710462
2000 Hamburg 71



Bernd Rudolph

Reflections-Werkstatt

Raytracing für jedermann

Raytracing. Klingt gut, wird sich der eine oder andere sagen. Doch was um Himmels Willen ist das? Und wie macht man das? Auf alle Fragen dieser Art soll dieser Workshop, der Ihnen die Welt der Reflexionen und Spiegelungen näher bringen will, Antwort geben.

Mit der Entwicklung immer schnellerer Prozessoren, die immer größer werdende Speicher zur Verfügung haben, steigen die Möglichkeiten der Grafikgestaltung enorm. Das wird besonders deutlich, wenn man einen VC20, einen C64 und einen Amiga vergleicht. Daß bei dieser Fortentwicklung natürlich auch die Ansprüche steigen, versteht sich. Mit den großen namhaften "Malprogrammen" ist es heute kein Problem mehr, eine Grafik zu erstellen. Wer sich jedoch eingehender mit dieser Materie beschäftigt, wird bereits festgestellt haben, daß ein Bild erst dann realistisch wirkt, wenn man sehr viel mit Farbverläufen arbeitet, die den Lichtreflexionen verschiedener Materialien gerecht werden. Nehmen wir beispielsweise an, der Kotflügel eines Autos sei blaumetallic. Man kann davon ausgehen, daß er an jeder Stelle "gleich blau" ist. Wenn man jetzt seinen Umriß zeichnet und blau ausfüllt, erhält man einen blauen Klumpen, dessen Ähnlichkeit mit einem Kotflügel nur noch zu erahnen ist. Der Grund ist eindeutig: Wenn man ihn in natura betrachtet, wird das einfallende Licht an der Oberseite anders reflektiert als an der Vorderseite oder der Rundung. Will man diesen Effekt auch in seiner Grafik erzielen, muß man entweder Profi sein und wissen, wie und wo Reflexe plaziert werden müssen, oder ein Programm benutzen, das diese Arbeit übernimmt.

Solche Programme bedienen sich eines Verfahrens, das man "Raytracing" nennt. "Ray" bedeutet "Strahl" und "tracing" "verfolgen". Ganz einfach gesagt, macht es nichts anderes, als eine Lichtquelle zu simulieren und die von ihr ausgehenden Strahlen

Kursfahrplan

Teil 1: Grundsätzliche Bedienung

Teil 2: Material und Licht

Teil 3: Werkstoffe und Oberfläche

Teil 4: Reflections in der Praxis

in der Richtung des Objektes zu verfolgen. Dabei wird berechnet, ob der Strahl irgendwo absorbiert wird, ob er zum Betrachter hin reflektiert wird und wohin der bestrahlte Körper seinen Schatten wirft. Daß solch eine Berechnung sehr kompliziert ist, dürfte deutlich geworden sein. Deshalb ist Raytracing nichts für diejenigen, die es eilig haben: Je nach Größe und Komplexität einer Grafik ist mit einer Rechenzeit zwischen einer halben und zehn Stunden zu rechnen. Das Warten wird jedoch fast immer belohnt!

Reflections

Seit einiger Zeit ist ein hervorragendes Programm auf dem Markt, mit dem Raytracing-Grafiken erzeugt werden können. Es heißt Reflections und bietet dem Hobby-Grafiker ein nahezu unerschöpfliches Reservoir an Funktionen und Gestaltungsmöglichkeiten zu einem erschwinglichen Preis. Wie man sich sicher vorstellen kann, ist das Programm sehr komplex und bedarf einiger Gewöhnung.

Wenn man Reflections aufruft, kommt man zuerst mit dem Manager in Kontakt. Von hier aus kann man alle Funktionen, die nacheinander auszuführen sind, bequem erreichen. Die Basis ist das Programm »Construct«, denn es dient der Erstellung der Szenen mit den darin enthaltenen Körpern sowie der Einstellung der Kameraposition und der Lichtverhältnisse. Beim Aufruf will »Construct« wissen, für wie viele Punkte, Objekte und Farben Speicher reserviert werden soll. Die Voreinstellung reicht für anfängliche Übungen auf jeden Fall aus und beansprucht den Speicher nicht zu sehr. Wir wollen uns zunächst nur mit dem Erstellen von Körpern beschäftigen.

Reflections kennt nur Dreiecke und Kugeln. Das ist aber nicht weiter schlimm, da »Construct« die von Ihnen eingegebenen Körper selbst in die Grundobjekte zerlegt. Sie haben es also nicht nur mit Dreiecken und Kugeln zu tun, sondern beispielsweise mit einem Tisch, einem Sofa oder einer Sektschale, die natürlich wieder aus einer Summe von Dreiecken oder Kugeln als Grundkörper bestehen. Zur Konstruktion solcher Körper stellt uns Reflections im Tools-Menü einige Hilfsfunktionen zur Verfügung.

Die erste davon ist »Rotkörp«. Mit ihrer Hilfe werden wir jetzt versuchen, eine Sektschale zu kreieren. Ein Rotationskörper entsteht, wenn sich zum Beispiel ein Rechteck um seine Längsachse dreht. In diesem Fall wäre es dann ein Zylinder. Was wir jetzt eingeben, ist nichts anderes als die Silhouette des Rotationskörpers. Wählen wir »Rotkörp« einmal aus und fahren mit der Maus in das rechte Bildschirmdrittel, genau auf die blaue Rotationsachse. Dort drücken wir die linke Maustaste und bewegen den Mauszeiger senkrecht fünf Kästchen nach oben, Maustaste, zwei nach links, vier nach unten, Maustaste, zwölf nach links, Maustaste, fünf nach links, sechs nach oben, Maustaste. Wollen Sie noch weiter Linien ziehen, die mit den vorherigen nicht in Verbindung stehen sollen, so können Sie einen neuen Anfangspunkt mit der rechten Maustaste setzen, wenn nicht, klicken Sie »Fertig« an. Nach jeder Zeichenoperation werden Sie gefragt, ob Sie zufrieden sind. Wenn Sie dies nicht bestätigen, können Sie nochmals beginnen, den Körper zu kreieren. Wir sind jedoch zufrieden mit unseren bisher erzielten Ergebnissen.

'Construct' fragt uns als nächstes nach der Anzahl der Winkelschritte, die wir durch Rechts-links-Bewegen der Maus einstellen (Wert erscheint im Message-Fenster). Je größer die Anzahl der Winkelschritte, um so besser wird die Rundung (Normalwert zwischen 10 und 20). Zuletzt werden Sie noch nach dem Namen des Körpers gefragt. Wir geben ihm einfach den Namen »Sektschale«. Unter diesem Namen können wir ihn dann künftig ansprechen.

Wählen wir im Tools-Menü den Menüpunkt »Zylinder«, erscheint auf dem Bildschirm ein Quadrat, das wir durch einfaches Bewegen der Maus verschieben können. Halten wir dabei die linke Maustaste gedrückt, können wir die Größe des Rechteckes bestimmen. Prinzipiell ist diese Funktion ein Spezialfall von »Rotkörp«, denn aus dem Rechteck, das wir definiert haben, wenn wir zum Schluß die rechte Maustaste drücken, entsteht durch Rotation ein Zylinder. Ebenso verhält es sich bei der Funktion »Kegel«. Die Funktion »Quader« ist zwar genauso zu bedienen wie »Zylinder«, nur mit dem Unterschied, daß man diesmal einen Block erhält, den man zwar in Höhe und Breite, nicht aber in der Tiefe verändern kann. Die Funktion »3D-Polygon« ist ganz ähnlich zu bedienen wie »Rotkörp«, bei einem Polygon müssen allerdings alle Punkte in Verbindung stehen und die Linien dürfen sich nicht überschneiden. Mit der linken Maustaste setzt man Punkte und mit der rechten beendet man den Vorgang. Dabei wird der zuletzt gesetzte Punkt automatisch mit dem ersten verbunden. Daraufhin will »Construct« wissen, ob Sie in das Polygon Löcher einfügen wollen, was Sie beliebig tun oder lassen können, vorausgesetzt, Sie fügen die Löcher wirklich in das Polygon ein und überschneiden nicht den Rand. Abschließend müssen Sie sich dann noch entscheiden, ob das Polygon zwei- oder dreidimensional sein soll und eventuell durch Bewegen der Maus mit anschließendem Druck auf die linke Taste die Tiefe einstellen.

Komplott zum Nachtisch

Jetzt bleibt nur noch die Funktion »Schnittk.« zu erklären. Dazu ein Beispiel: Stellen Sie sich einmal einen Fußball vor, den Sie in 100 Scheiben zerschnitten haben. Jede dieser Scheiben ist ein Kreis mit unterschiedlichem Durchmesser. Bei der Funktion »Schnittk.« würden Sie genau umgekehrt vorgehen. Zuerst würden Sie lauter unterschiedlich große Kreise erstellen, die dann hintereinandergelegt wieder den Ball ergeben. Allerdings wäre es unsinnig, mit »Schnittk.« Kugeln zu erstellen, da es dafür ja das Menü »Kugel« gibt. Was Sie hier eingeben müssen, sind Polygone. Vor jeder Eingabe will »Construct« wissen, in welcher Tiefe das Polygon, das Sie als nächstes eingeben wollen, liegen soll. Die Einstellung erfolgt wie immer mit der Maus. Zur Orientierung erscheinen die einzelnen Polygone in verschiedenen Farben und Strichmustern. Zum Schluß werden die Polygone automatisch der Tiefe nach sortiert.

Soviel zur Erzeugung von Körpern. Wenden wir uns nun ihrer Darstellung zu. Die nötigen Funktionen finden wir im Menü »Plotten«. Der erste Menüpunkt dient der Definition des zu plottenden Körpers. Nachdem er angewählt wurde, erscheint ein Fenster, in dem alle Körper aufgelistet sind. Durch das Bewegen der Maus in vertikaler Richtung und Bestätigen mit der linken Maustaste kann man den Körper bestimmen, auf den sich die Ausgabe beschränken soll. Mit »Fenster« kann man auf die gleiche Weise einen Körper fensterfüllend darstellen. Die nachfolgenden sechs Menüpunkte beziehen sich auf die Blickrichtung und sind vermutlich keiner näheren Erklärung bedürftig.

»Bild*2« vergrößert das momentane Bild nicht etwa, wie man vermuten könnte, sondern verkleinert es auf die Hälfte. Dies ist oft dann notwendig, wenn man später die Position der Kamera sowie die der Lampen überprüfen oder verändern will. Mit Hilfe der Funktion »Persp« kann man sich die Szene aus der Kameraposition ansehen. Wundern Sie sich nicht, wenn die eben erstellte Szene plötzlich unheimlich winzig erscheint, denn das liegt an der Einstellung der Kamera, doch dazu später. »Plotten« dient dazu, einen zuvor mit »Plotkörp« ausgewählten Körper zum bestehenden Fensterinhalt hinzuzufügen. Die folgenden fünf Menüpunkte beziehen sich auf die Darstellung der Körper.

»Hidd0« veranlaßt die Darstellung aller Dreiecke, »Hidd1« die Darstellung der Dreiecke, die dem Betrachter zugewandt sind, und mit »Hidd2« werden die rückseitigen Dreiecke gepunktet dargestellt. »Farb1« ist die Grundeinstellung, in der alle Körper die gleiche Farbe haben, während mit »Farb2« erreicht wird, daß jeder Körper entsprechend seinem Material eine Farbe zugeordnet bekommt. Ab einer gewissen Anzahl an Materialien wiederholen sich die Farben natürlich.

Als letztes folgt die Kameraeinstellung, um die wir uns in der nächsten Folge kümmern werden. Dann werden wir uns mitten unter die Dreiecke stürzen und beginnen, eine kleine Szene zu kreieren. Bis dahin empfehle ich Ihnen, einmal die eben besprochenen Funktionen auszuprobieren, um auszutesten, wie sie anzuwenden sind, und was man mit ihnen erreichen kann.

(hs)

Ingmar Reyer

Amiga Common Tex

Power without the price?

Tex ist ein weit verbreitetes Satzprogramm, das auf nahezu allen Rechnertypen und Betriebssystemen verfügbar ist. Common-Tex auf dem Amiga muß nun zeigen, ob auch auf diesem Rechner ein professionelles Arbeiten möglich ist.

Schon seit längerer Zeit geistert der Mythos eines Schriftsatzprogramms durch die Amiga-Szene. Es soll für die Erzeugung wissenschaftlich-technischer Texte in Buchdruckqualität und professioneller Dokumente noch besser geeignet sein als alle bekannten Desktop-Publishing- und Textverarbeitungsprogramme. Das Programm, von dem hier die Rede ist, heißt Tex (sprich: Tech).

Geschichtliches

Tex wurde von Donald E. Knuth von der Stanford University Mitte der 70er Jahre entwickelt. Seit dieser Zeit hat Tex eine weite Verbreitung gefunden und ist ständig weiterentwickelt worden. Es ist inzwischen für fast alle Rechnertypen und Betriebssysteme erhältlich, angefangen vom Großrechner über UNIX-Workstations bis hin zum PC und Amiga.

Bei Amiga-Common-Tex handelt es sich um eine komplette PD-Implementation der Common-Tex-Version für UNIX-Rechner. Dabei ist die Amiga-Version aber keine abgespeckte, sondern sie bietet all das, was auch unter UNIX machbar ist – und das ist nicht wenig. Das Programmpaket besteht aus vier Disketten, die das eigentliche Tex-Programm, einen Previewer, Druckertreiber und viele Zeichensätze beinhalten.

Die Erstellung eines Dokuments stellt sich bei Tex etwas anders dar als bei herkömmlichen Textverarbeitungsprogrammen. Tex ist im Grunde genommen eine Programmiersprache. Wie bei einem C-Programm wird ein Text mit Befehlen übersetzt und anschließend mit Hilfe eines Treibers auf einem Drucker ausgegeben. Der Befehlssatz von Tex umfaßt rund 300 Basisbefehle, aus denen weitere 600 Makrobefehle zur Verfügung stehen. Allein daraus wird ersichtlich, daß der Anwender ohne ein geeignetes Lehrbuch nicht auskommen kann.

Tex, das Abenteuer

Leider ist die mitgelieferte Anleitung von Common-Tex nicht ausreichend, um den Wissensdurst des Anwenders zu befriedigen.

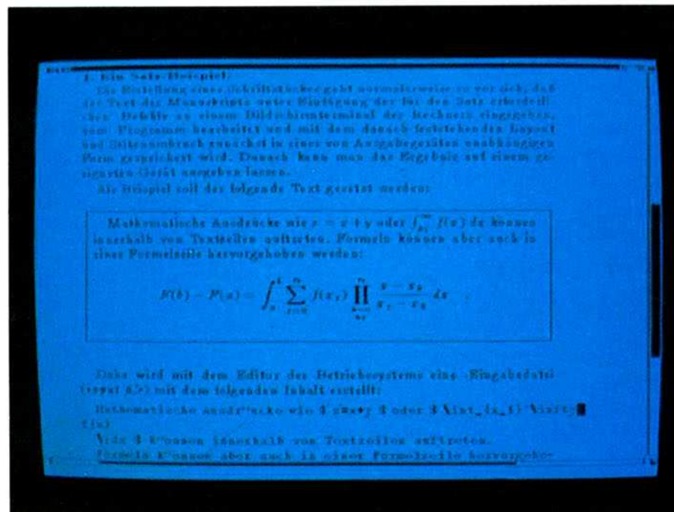


Bild 1. Der Previewer von Common-Tex ermöglicht das Überprüfen eines Textes vor dem Ausdruck

Nach dem Erstellen eines Programms mit einem Texteditor kann man es von Tex compilieren lassen. Tex erzeugt danach, wenn kein Fehler gefunden wurde, ein DVI-File. Dieses File ist ein geräteunabhängiges Ausgabe-File, das mit dem Previewer auf dem Bildschirm oder mit einem Druckertreiber auf Ihrem heimischen Drucker ausgegeben werden kann. Hat Tex beim Compilieren einen unbekannten Befehl entdeckt oder fehlt ihm ein Zeichensatz, der vom User im Quelltext angewählt wurde, dann gibt Tex ausführliche Fehlermeldungen aus, und der Anwender muß

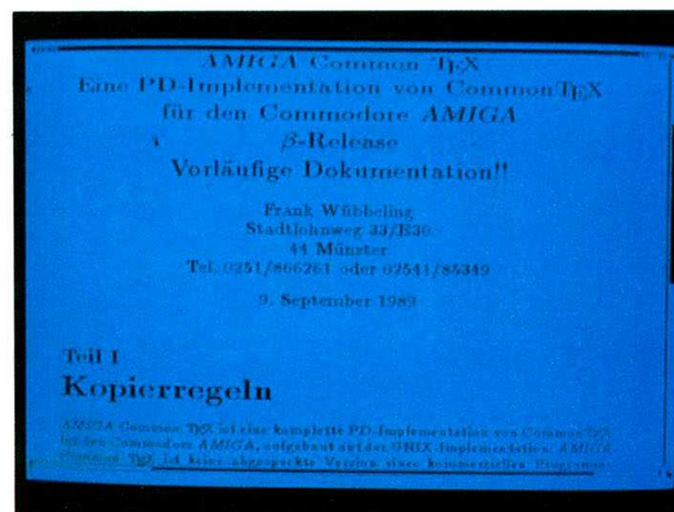


Bild 2. Selbst komplizierte Texte stellen für Common-Tex überhaupt kein Problem dar

seinen geschriebenen Text noch einmal korrigieren und erneut compilieren.

Die Fonts sind die herausragende Stärke des Amiga-Common-Tex-Pakets. Die Auflösung ist besonders klar und leserlich.

Fonts sind das A und O

Zeichensätze stehen in verschiedenen Größen und Auflösungen zur Verfügung. Die Ausgabe eines Textes mit einem 24-Nadeldrucker und einer Auflösung von 360 dpi geht dabei recht flott und gestochen scharf vonstatten, obwohl die Textseiten im Grafik-Modus des verwendeten Druckers zu Papier gebracht werden.

Spezialitäten von Tex

Tex besitzt im Textsatz viele Fähigkeiten wie Randausgleich, Zeilenumbruch, automatische Trennung, Fußnoten, Erstellung von Inhaltsverzeichnissen, Proportionalschrift, Tabellensatz, Akzente und viele Sonderzeichen.

Besondere Erwähnung verdient der Formelsatz. Mit seiner Hilfe können komplexe Formeln innerhalb von Textzeilen oder auch abgesetzt vom Text erscheinen. Die Darstellung von langen Brüchen oder Wurzelausdrücken stellt überhaupt kein Problem für Tex dar.

Wer jetzt meint, daß Tex für ihn genau das richtige sei, aber nicht glaubt, ein solch komplexes Programm jemals verstehen zu können, der kann mit einem Beispiel eines Besseren belehrt werden.

Der Ausdruck »special filename xdim ydim« führt dazu, daß ein IFF-Bild an der Stelle, wo der Befehl steht, mit der in »xdim« und »ydim« angegebenen Größe im Dokument ausgegeben wird.

Erst gucken, dann drucken

Zugegeben, die vielen Befehle wollen zuerst einmal erlernt sein, aber sobald man einen gewissen Grundwortschatz besitzt, geht die Arbeit mit Tex schnell von der Hand.

Der Previewer des Common-Tex-Pakets ist besonders gut gelungen. Er stellt die von Tex erzeugten DVI-Files gra-

fisch auf dem Bildschirm dar (WYSIWYG-Darstellung), und der User kann sich ein Bild von seinem Dokument machen, ohne jedesmal viel Papier und Zeit zu verschwenden. Viele ältere Tex-Implementationen auf anderen Rechnern bieten dieses komfortable Tool nicht.

Was wird an Hardware benötigt?

Ein so komplexes Programmpaket wie Tex besitzt natürlich größere Hardware-Anforderungen. Aber mit Common-Tex kann man schon arbeiten, wenn man 1 MByte Speicher und zwei Diskettenlaufwerke besitzt. Bequem wird es allerdings erst, wenn der Anwender im Besitz einer Festplatte ist. Dann fallen die häufigen Diskettenwechsel und die längeren Ladezeiten weg. Ach, beinah hätte ich vergessen zu erwähnen, daß außerdem ein Drucker vorhanden sein muß (kleiner Scherz am Rande).

Das Urteil

Amiga-Common-Tex stellt ein einmaliges Werkzeug für die Verarbeitung von naturwissenschaftlichen Texten und langen Dokumenten dar. Außerdem ist es möglich, Makrobefehlserweiterungen als PD nachzukaufen. Mit deren Hilfe wird Tex befähigt, Musikpartituren oder chemische Formeln zu setzen. Kein anderes textverarbeitendes Programm ist so flexibel und vielseitig. Somit stellt dieses Programm einen Stern am PD-Himmel dar, den sich kein interessierter Anwender entgehen lassen sollte.
Wer allerdings nur hin und wieder einen Brief oder eine Einladung schreiben will, sollte sich mit einer normalen Textverarbeitung begnügen. Dazu ist die Einarbeitungsphase in Common-Tex zu beschwerlich und langwierig.

(vb)

Name: Amiga Common Tex
Quelle: Wolf Computertechnik

Kurz vor Redaktionsschluß erreichte uns die Meldung, daß in nächster Zeit mit einer neuen Version gerechnet werden kann.

Die PD-Werkzeugkiste

Der PD-Bereich ist vollgepackt mit einer Vielzahl guter Utilities, kompakter Programme und spannungsgeladener Spiele. Wir stellen Ihnen mit Angel-Paint ein gutes Grafikprogramm vor. Daneben wird die Werkzeugkiste mit der MIDI-Serie und einem Beitrag über TEX vervollständigt.

Es wird für die kommerziellen Programme immer schwieriger, ihren preislich günstigeren Konkurrenten aus der PD-Branche die Stirn zu bieten. Ein Grund dafür ist, daß sich die PD-Programme in ihrer Leistungsfähigkeit kaum noch von den kommerziellen Programmen unterscheiden. Nur der Preis differiert stark.

Share Nr. 9 – D-Paint oder Angel-Paint?

Mit Angel-Paint präsentiert sich eine Alternative zum bisweilen bewährten D-Paint. Der Bildschirm wird zum größten Teil von der Malfläche ausgefüllt. Lediglich am rechten Bildschirmrand erstreckt sich eine Werkzeugkiste und am oberen Bildschirmrand ein Pull-Down-Menü, über das sich alle Funktionen des Programms steuern lassen. Diese Werkzeugkiste stellt sich wie folgt dar:
Dem Benutzer zeigen sich eine Vielzahl von Symbolen, die über die Art des Zeichnens Auskunft geben. So bedeutet das Kreis-Symbol die Formung eines Kreises, wobei nur noch der Radius und der Bestimmungsort per Maus angezeigt werden muß. Des weiteren lassen sich Ellipsen, Rauten beziehungsweise Polygone erstellen. Zusätzlich kann die Art des Striches variiert werden, wobei Ihnen das Programm die Möglichkeit offeriert, die Dicke des Zeichenstriches zu verändern. Daneben können mit einem Strich zwei oder gar drei Linien gezeichnet werden. Einzelne Flächen lassen sich ebenso problemlos ausfüllen. Eine umfangreiche Farbpalette, die über die Farbnuancierung des Stiftes, der einzelnen Objekte und des Hintergrundes entscheidet, garantiert eine schönere Gestaltung.
Damit der Benutzer jederzeit über seine gegenwärtige Position informiert ist, werden die Daten für die X/Y-Koordinaten am unteren Bildschirmrand angezeigt. Man sollte meinen, daß nun sämtliche Optionen des Programms abgehandelt wären, aber weit gefehlt. Bisher wurde lediglich auf die Werkzeugkiste eingegangen. Beim Pull-Down-Menü ergeben sich noch einige interessante Aspekte: Das Projekt-Menü wartet mit bereits bekannten Optionen wie Bild-Laden etc. auf. Zusätzlich wird Ihnen die Freiheit gelassen, das soeben erstellte Bild gleich zum Drucker zu schicken. Beim Pinsel-Menü mit seinen unterschiedlichen Optionen werden direkt die gemalten Objekte angesprochen. Eine Rotationsoption sorgt dafür, daß ein vorher definierter Pinsel um einen entsprechenden Winkel rotieren kann. Zusätzlich läßt sich ein vorher definierter Pinsel an der X/Y-Achse spiegeln.
Hinter dem etwas allgemein formulierten Begriff "Diverses-Menü" verbirgt sich eine Vielzahl nützlicher Optionen. So lassen sich beispielsweise Flächen mit einem vorher definierten Muster ausfüllen. Nach dem Erstellen eines Kreisausschnitts werden nach Aktivierung der Bogen-Option die Endpunkte des Ausschnitts mit dem Kreismittelpunkt durch Geraden verbunden, so daß eine Art Kreisdiagramm entsteht. Die unterschiedlichen Schriftarten können über das Schrift-Menü eingestellt werden. Das Werkzeug-Menü verfügt nur über die Option zum Ein- und Ausschalten der Werkzeugkiste. Es versteht sich von selbst, daß alle Bilder, die mit D-Paint erstellt wurden, auch mit Angel-Paint ladefähig und modifizierbar sind. Mit Angel-Paint stellt sich ein recht gelungenes und kompaktes Programm vor.

(Jürgen Seibel/vb)

Name: Share Nr. 9
Quelle: Herrmanns & Kommelter



Bild 1. Leichtes Arbeiten mit Angel-Paint

Neue PD-, Share- oder Freeware-Serien für den Amiga schießen wie Pilze aus dem Boden. Neben bekannten Reihen, wie zum Beispiel die Fish- oder Kick-PDs, versuchen auch neue Serien, sich einen Namen auf dem nicht nur bei Usern beliebten Markt zu machen.

UGA Disk 1

So auch eine neue Reihe aus den Niederlanden, die sich hauptsächlich auf Utilities für den Computeranwender bezieht. Ob es ihr gelingt, sich in die Reihe der "Großen" zu stellen, wird sich zeigen. Auf den ersten Blick macht die

vollgepackte Disk einen guten Eindruck. Die Programme sind alle über Tastenkombinationen zu erreichen, man muß sich endlich nicht mehr durch etliche Windows bis zum eigentlichen Programm durchkämpfen.

Vom Programmierer bis zum "Möchtegern-Crack" ist für jeden etwas dabei. Um Disketten effektiver auszunutzen, ist auf der Diskette ein **»Cruncher«** enthalten, der Programme, sollten sie noch nicht gepackt sein, um ungefähr die Hälfte verkürzt. Auch ist gleichzeitig ein Recruncher integriert, der gepackte Files, sofern sie mit diesem Packer "gecruncht" wurden, wieder entpackt und sie auf die ursprüngliche Länge bringt.

Für grafiktalentierte Amiga-User gibt es wieder einen **»Sprite-Editor«**, der durch ausgetüftelte Programmierung und Menüführung besticht. Bewegungsabläufe der Sprites lassen sich festlegen; ein Animationsprogramm ist ebenfalls vorhanden. Alle Funktionen, wie beispielsweise das Setzen der Farben, lassen sich durch Tastenkombinationen oder durch Menüs anwählen. Für Benutzerfreundlichkeit ist also gesorgt.

Fraktalgrafik auf dem Amiga wird immer beliebter. So befindet sich auch auf dieser Disk ein Fraktalprogramm, **»MandelVRoom«**, das Apfelmännchen und "Genossen" auf den Bildschirm bringt, die dann durch Zusatzfunktionen beliebig verändert werden können. Allerdings wird dieser Spaß durch ziemlich lange Rechenzeiten getrübt. Wen dies aber nicht stört, der wird mit diesem Programm mehr als zufrieden sein.

Wer auf Grafikklau gehen oder sich "mal eben die neuesten Chars aus einem Intro" holen möchte, der ist mit dem **»Searcher«** bestens bedient. Der Speicher kann mittels Cursortasten in vier Richtungen durchscrollt werden. Wird man fündig, kann die Bitplane als IFF-File abgespeichert werden. Bei "mehrplanigen" Grafiken ist es möglich, Bitplane für Bitplane hintereinander abzuspeichern und sie hinterher, zum Beispiel mit dem Butcher, wieder zusammenzusetzen. Die übersichtliche Aufmachung rundet den positiven Eindruck ab.

Bleiben wir noch ein bißchen beim Thema Grafik. Mit dem



Bild 2. Mit »BOOT'EM« lassen sich Programme in den Boot-Block einbinden

Utility **»IconLab«** lassen sich Icons, die normalerweise auf einem Workbench-Screen zu sehen sind, anschauen, katalogisieren und editieren. Wer häufig von der Workbench lädt, wird dieses Programm zu schätzen wissen, da sich Icons leicht nachbearbeiten lassen.

Der CLI läßt so manchen Einsteiger oft nicht ungestraft, werden doch Befehle – unsachgemäß ausgeführt – oftmals ignoriert und mit irgendwelchen Fehlermeldungen quittiert. Der **»CLI-Wizard«** macht Schluß mit dem lästigen Blättern im Handbuch; die meisten CLI-Befehle sind mittels Gadgets zu erreichen. Auch einzelne Files lassen sich mit diesem Tool kopieren.

Für Leute, die für ein dBase-Original nicht mehrere hundert Mark bezahlen wollen, gibt es auf der UGA-Disk eine preiswertere Möglichkeit – mit **»DBase-Wizard«** erhalten Sie eine zwar abgespeckte und dennoch nützliche Adreßverwaltung.

Kommen wir jetzt zu den unscheinbareren Utilities. Wer schon immer mal einen "flotten Boot-Scroller" programmieren wollte und bisher vor verschlossenen Registern stand, dem sei mit dem **»Bootwriter«** geholfen. Mit ihm lassen sich Boot-Intros auf einfache Weise generieren und abspeichern. Das Programm **»BOOT'EM«** setzt Programme in den Boot-Block, das heißt, das Programm wird gleich nach einem Reset geladen. Dabei können die Programme auch länger als 512 Bytes, die eigentliche Länge eines Boot-Blocks, sein.

Zu einem anständigen Intro gehören natürlich auch Sprites, die sich am besten noch

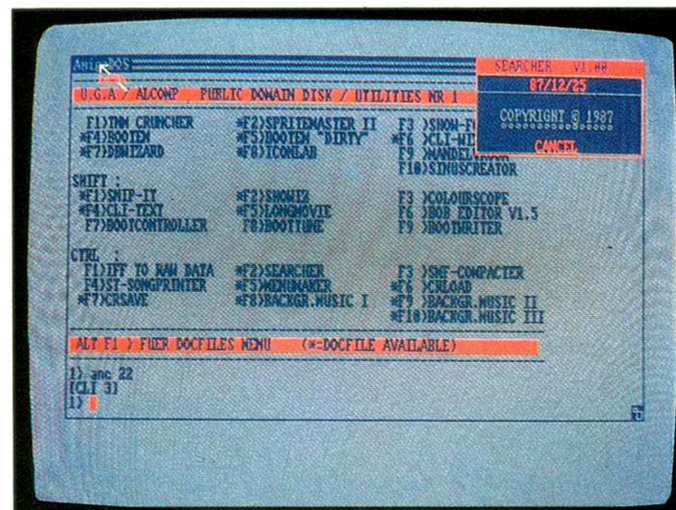


Bild 3. »Searcher« – ein Bitmap-Analyzer der Extraklasse

in bizarren Bahnen über den Screen bewegen. Mit dem **»Sinus-Creator«** ist auch dies kein langer Rechenaufwand mehr. Bestimmte Formen sind per Tastendruck aufzurufen, wem diese aber nicht gefallen, kann selbst sein Glück mit der Maus versuchen. Mit dem Programm **»Bob-Editor«** lassen sich Blitterobjekte entwerfen und editieren. Allerdings ist keine Lupenfunktion beim Editieren vorhanden, was sicherlich wünschenswert wäre, da sich Bobs ohne Lupe nur schwer erstellen lassen. Hat man ein Bob fertiggestellt, kann man es auf Disk als C-Source abspeichern.

Bei der Arbeit mit gesampelten Sounds wirkt der IFF-Header, der Informationen über das Sample enthält, oft sehr störend. Da der Computer diesen Header normalerweise nicht erkennt, wird er beim Abspielen des Samples als Knacken oder ähnliches hörbar. Um diesem entgegenzuwirken, befindet sich auf der UGA-Disk 1 ein **»IFF-Konverter«**. Er sucht, nachdem ein Sample eingeladen worden ist, nach diesem Header und entfernt ihn. Das Sample kann dann als Raw-Format abgespeichert werden. Allerdings ist es danach nicht mehr möglich, das Sample weiterzubearbeiten (beispielsweise mit Audiomaster II), da dem Programm ja nun die nötigen Informationen über das Sample fehlen. Deshalb sollte man sich von den Samples immer eine Sicherheitskopie machen, bevor man sie in ein Raw-File konvertiert.

Mit dem Programm **»Background-Music«** lassen sich Samples unabhängig im Hintergrund abspielen, während der Prozessor beispielsweise mit dem Abarbeiten einer Adreßverwaltung beschäftigt ist. Um eine hohe Kompatibilität zu gewährleisten, sind auf der Diskette gleich drei Versionen enthalten.

Diese Diskette ist vollgepackt mit nützlichen Utilities und läßt keine Wünsche offen. Wenn die UGA-Reihe dieses Niveau durchhält, dann hat sie gute Chancen, sich einen sicheren Platz auf dem PD-Markt zu verschaffen.

(Timo Siebert/vb)

Name: UGA Nr. 1
Quelle: APS Electronic



Rund um das Thema MIDI geht es bei dieser PD-Serie. Hier findet bestimmt jeder Musikfan etwas

Andreas Polk

Die MIDI-Story

MIDI – ein Schlagwort in der modernen Musikerstellung. Leider sind die Informationen zu diesem Thema noch sehr rar, für den Einsteiger oft nur schwer verständlich und sehr theoretisch.

Die MIDI-PD-Serie möchte den MIDI-Interessierten den Einstieg in die MIDI-Welt erleichtern und fortgeschrittenen Anwendern einige nützliche Dinge in die Hand geben. Ein Grund, diese Reihe einmal näher unter die Lupe zu nehmen.

MIDI steht für Musical Instruments Digital Interface. Digitale Schnittstelle für Musikinstrumente steht also für ein einheitliches Datenübertragungs- und Datenaustauschsystem im Bereich der Computer-Musik.

Sie umfaßt zur Zeit 16 Disketten. Ins Leben gerufen wurde sie von Oliver Wagner. Er möchte interessierten Einsteigern genauso weiterhelfen wie fortgeschrittenen Usern. So befinden sich nicht nur Utilities auf den Disketten, sondern auch Texte, in denen Grundlagen erklärt werden, und Bilder, die den Bau eines MIDI-Interfaces erläutern. Außerdem gibt es noch Texte, die als Hilfe für Neulinge gedacht sind sowie eine Rubrik mit dem Namen »Fragen & Antworten«. Diese Texte haben kein bestimmtes Themengebiet.

Mit MIDI Musik machen

Hier reicht es vom schon erwähnten MIDI-Kurs bis hin zu Kaufhinweisen und sonstigen Erläuterungen zum Thema MIDI. Neben diesen einsteigerfreundlichen Dingen gibt es natürlich viel Interessantes für den fortgeschrittenen Anwender. So finden sich auf den Disketten Utilities, wie beispielsweise ein Echo-Generator oder Bankloader. Wer allerdings damit rechnet, einen vollständigen Sequencer à la KCS oder Texture auf diesen Disketten zu finden, wird enttäuscht. Aber – wer kann das schon erwarten?

Ein Problem im Bereich MIDI ist natürlich, daß es nicht nur einen Synthesizer gibt, sondern eine Menge an unterschiedlichen Geräten. Die meisten Utilities beziehen sich auf den preisgünstigen Casio-Synthesizer der CZ-Reihe. Aber auch Dinge für den DX-7 oder Geräte von Roland und Kawai sind zu finden. Hier hängt es also von den Anwendern ab, in welcher Weise sie diese Serie unterstützen. Nur so kann sie weiterwachsen, und mehr Geräte können unterstützt werden. Doch nicht nur für MIDI-Anwender ist diese Serie interessant. Neben den vielen Soundbänken sind sehr viele gesamplete Sounds auf den Disketten abgespeichert. So befindet sich auf der Diskette Nr. 2 das gesamte Drumset des Roland TR-505 Drumcomputers. Sie können so mit dem KCS, aber auch mit anderen Soundeditoren wie beispielsweise Dynamicdrums oder TFMX benutzt werden.

Wer also Tools oder Hinweise zu seinem MIDI-Equipment sucht, ist mit dieser Serie gut beraten. Oder wollen Sie nur mal in das Thema MIDI reinschnuppern? Bitte, hier können Sie das günstig auf 16 Disketten tun!

(vb)

Name: MIDI-Serie
Anbieter: APS-Electronic

Nr.	Name
1	div. Tools
2	CZSPLIT +
3	Dump-Utility für Roland S-220
7	Bank-Loader für CZ 101/1000/3000
12	Effektprg. für Sampler
12	Editor für MT-32
14	Editor für CZ 101/1000/3000
14	Bankloader für D110 und S220

Utilities

Nr.	Name
2	Drumset Roland TR-505
3	Banks für DX-7
4	Sounds für CZ-1
4	Bank für MT-32
8	Banks für MT-32
8	Samples
10	Samples von RX-5 und CZ 101
11	Sounds und Banks für CZ 101
12	Sounds für CZ 101
13	Banks für K1
14	Bank für MT-32
14	Soundbanks für FB-01

Banks und Sounds

Nr.	Name
4	D-50 Editor
16	KCS
16	Editor MT-32

Demos von kommerziellen Programmen

Nr.	Name
7	midi.libraryV2.0
8	KCS Drumpatterns
12	Texte
13	Texte
15	Texte

Sonstiges

Eine kleine Übersicht der Vertreiber von Public Domain, Free- und Shareware sowie Prüf-vor-Kauf-Programmen (ohne Anspruch auf Vollständigkeit)

A.P.S. Electronic
Sonnenborstel 31
3071 Steimbke
Tel.: 05026/1700

Bavariansoft
Friedrich Neuper
Postfach 72
8473 Pfreimd
Tel.: 09606/286

Belilingrath, Christian
Hans-Böckler-Str. 55
5860 Iserlohn
Tel.: 02371/24192

Digital Marketing
D. Mückter
Krefelder Str. 16
5142 Hückelhoven-Baal
Tel.: 02435/2086,428, oder 1295

Dombrowski Rüdiger
Postfach 71 04 62
2000 Hamburg 71
Tel.: 040/6428225

Herrmanns & Kommmelter
Vom-Bruck Platz 45
4150 Krefeld 1
Tel.: 02151/399833

Hieske, Dieter
Schillerstr. 36
6700 Ludwigshafen
Tel.: 0621/673105

Keim,Peter
Vogelsanger Str. 34
5000 Köln 30
Tel.: 0221/520765

Maxon Computer GmbH
Industriestr. 26
6236 Eschborn
Tel.: 06196/481811

Nordsoft
Schwenecker & Behnke
Heidbergstr. 75
2800 Bremen 21
Tel.: 0421/611430

Pawlowski, Patrick
Software Service
Ellerbruch 19
2177 Wingst
Tel.: 04778/7294

Software Empire
Lange Str. 118
2880 Brake (Utw)
Tel.: 04401/71348

TechnikSupport Verlag GmbH
Bundesallee 36-37
1000 Berlin 31
Tel.: 030/8621314

Wolf Computertechnik
Deipe Stegge 187
4420 Coesfeld
Tel.: 02541/2874

Österreich

Küppers, Bernd
Felberstr. 7
A-5730 Mittersill
Tel.: 06562/282

M.A.R-Computershop
A-1100 Wien
Weldengasse 41
Tel.: 0222/621535

(vb)

Public-Domain-Spieleshow

Alle Programme, die es in der PD gibt, auf Disketten gebannt und übereinander gestapelt, das würde einen stattlichen Berg ergeben. Nicht selten haben PD-Vertreiber 5000 und mehr verschiedene Disketten auf Lager. Der Amiga-Benutzer darf sich freuen, denn in diesem Berg verborgen glänzen viele Software-Juwelen, die es nur zu finden gilt.

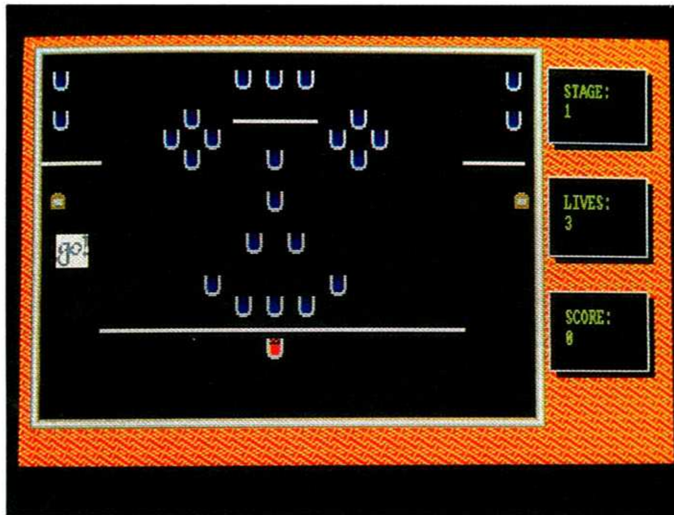
Trotz des gruseligen Namens handelt es sich bei diesem Spiel nicht etwa um ein Ballerspiel, sondern um ein Jump'n'Run-Spiel, in dem der Spieler die Rolle des grimmigen Sensenmannes übernimmt, also Gevatter Tod spielt. Dieser muß auch hier seiner gewohnten Tätigkeit nachgehen und in 20 verschiedenen Szenen verlorene Seelen einsammeln.

The Death

Jedoch wird die Aufgabe des Todes durch Horden kleiner, keulenschwingender Gnome erschwert, die in jedem Bild die Seelen bewachen und nur darauf warten, den Tod mit einem kernigen Hieb niederzustrecken. Außer den Keulenzwergen treiben sich aber auch noch andere Finsterlinge in den einzelnen Bildern herum. Da wuseln Schlangen umher, leuchtende Bälle gleiten durch die Luft und kleine Ku-Klux-Klan-Kuttenträger schleichen durch die Szenerie. Mitunter verwandeln sich die barschen Gnome für kurze Zeit in Smilies, die der Tod dann einsammeln kann.

Gelingt es, verschwinden für einen kurzen Moment alle anderen Untiere aus einem Level. Kleine Bonusbläschen, die in unregelmäßigen Abständen auftauchen, erleichtern die Aufgabe des Todes. Sie können von der Lähmung der Gegner bis zum Transport ins nächste Level alles mögliche enthalten. Ist es dem Tod gelungen, alle Seelen eines Levels zu erhaschen, geht es in der nächsten Stufe weiter.

Obwohl Grafik und Sound eher einfach gehalten sind, bringt "The Death" erstaunlich viel Spaß ins Spiel.



Der Tod muß Seelen sammeln, was gar nicht so leicht ist

Name: The Death
Enthalten auf: Kickstart 242
Vertrieb: siehe Anbieterliste

Ein Raumschiff ist auf dem fernen Planeten Jovi notgelandet. Die Besatzung muß nun versuchen, von dem unwirtlichen Felsklotz wieder zu entkommen. Zwar steht eine Fähre zur Verfügung, aber wie das Schicksal so spielt, gilt es, sie erst einmal durch diverse Höhlen zu manövrieren, bevor gestartet werden kann.

Escape from Jovi

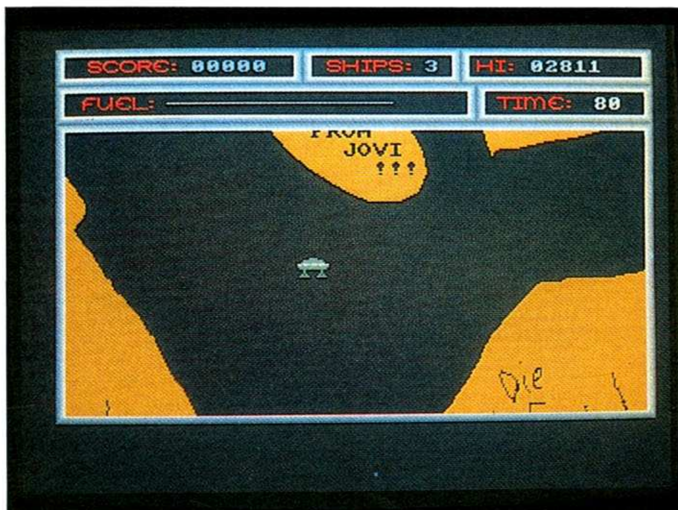
Dem Spieler fällt die verantwortungsvolle Aufgabe zu, die Fähre durch die verschlungenen Höhlen zu steuern. So einfach, wie es sich anhört, ist diese Aufgabe jedoch nicht zu bewerkstelligen, da die Fähre permanent der Anziehungskraft des Planeten ausgesetzt ist und schon die kleinste Berührung der Höhlenwände fatale Folgen hat. Sackgassen, Engpässe und verschlungene Korridore stellen das Geschick des Piloten auf eine harte Probe, die durch einen schnell abnehmenden Treibstoffvorrat nicht leichter wird.

Die Höhlen, durch die der Flug führt, sind weit größer als ein Bildschirm, und das Programm läßt die Szenerie am Betrachter vorbeiscrollen.

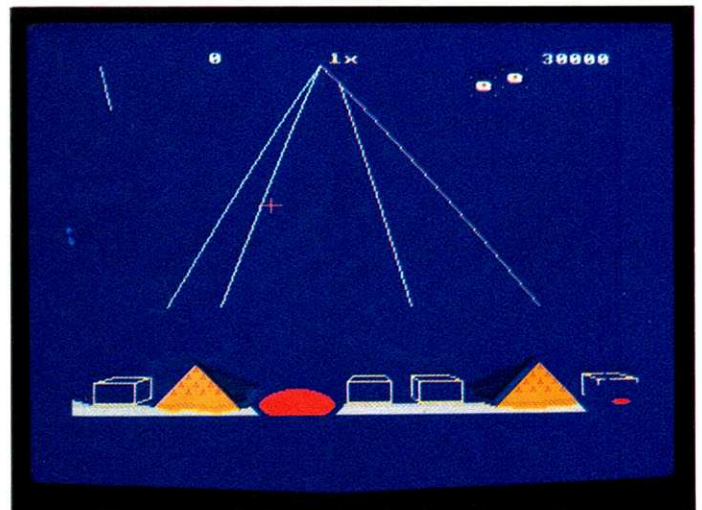
Escape from Jovi kann global als "Beschleunigungsspiel" bezeichnet werden. Tatsächlich beschränkt sich die Aufgabe des Spielers auf das Beschleunigen und das Lenken der Rettungsfähre. Escape from Jovi besticht durch saubere Programmierung und viele verschiedene Höhlen mit kniffeligem Layout. Eine speicherbare Highscore-Liste sichert die besten Spielergebnisse auf Diskette, und es macht viel Spaß, wieder und wieder zu versuchen, einen einmal aufgestellten Rekord weiter zu verbessern.

Bedenkt man, daß diese Spielthematik die Grundlage für eine ganze Reihe kommerzieller Spiele ist, braucht sich Escape from Jovi nicht zu verstecken. Was die Spielbarkeit anbetrifft, läßt dieses PD-Spiel so manches Kommerzprodukt hinter sich.

Name: Escape from Jovi
Enthalten auf: MDA 001
Vertrieb: siehe Anbieterliste



Im Kampf gegen die Gravitation



Verteidigen Sie Städte gegen anfliegende Raketen

Zwischen zwei verfeindeten Nationen ist ein bewaffneter Konflikt ausgebrochen, der mit ferngelenkten Raketen ausgetragen wird. Dem Spieler dieses actiongeladenen Programms fällt die verantwortungsvolle und schwere Aufgabe zu, das Mutterland eines der beiden Gegner vor den nahenden Raketen des Feindes zu beschützen. Zu diesem Zweck steht eine Kanone zur Verfügung, deren Geschosse in den Weg der anfliegenden Raketen gefeuert werden und dort explodieren.

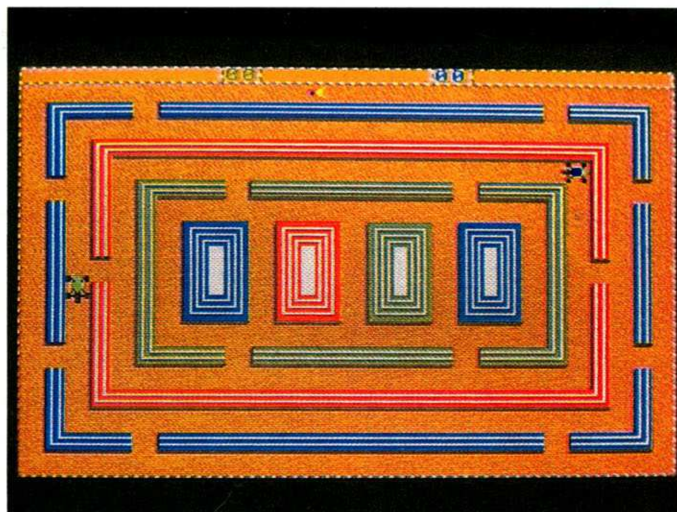
Missile

Jede Rakete, die in den Wirkungskreis einer solchen Explosion gerät, wird dadurch unwirksam und kann keinen weiteren Schaden anrichten.
Die Munition für die Kanone wird in Pyramiden gelagert, die wie die Städte geschützt werden müssen. Trifft trotz aller Bemühungen doch eine der Raketen das Munitionsdepot, werden alle Vorräte vernichtet und es kann nicht weitergefeuert werden.
Ab und an ziehen Spionagesatelliten ihre Bahn über den Kriegsschauplatz. Gelingt es, sie zu treffen, winkt ein saftiger Punktebonus.

Name: Missile
Enthalten auf: Taifun 6
Vertrieb: siehe Anbieterliste

Missile ist erstaunlich schnell und präzise in seinem Spielablauf. Schon nach wenigen Angriffswellen werden die ankommenden Raketen so schnell, daß der Spieler sich auf die Verteidigung einiger weniger, dicht beieinanderliegender Städte beschränken muß. Denn ist erst die letzte Stadt vernichtet, ist auch das Spiel beendet.

Slotcars basiert auf einer Spielidee, die Mitte der 70er Jahre erstmalig auf den damals aktuellen Telespielkonsolen realisiert wurde. Zwei Autos rasen durch ein Labyrinth. In den Wänden des Irrgartens wurden viele Lücken belassen, durch die die Wagen manövrieren können. Beide Autos haben nur ein Ziel: die Jagd aufeinander. Um sich gegenseitig zur Strecke bringen zu können, ist jedes Auto mit einer Kanone ausgestattet, die jedoch nur in Fahrtrichtung feuern kann.



Ein Autoduell im Labyrinth

Slotcars

Da es keine Bremsen gibt und beide Fahrzeuge unentwegt in Bewegung sind, muß schon einiges an Manövriergeschick bewiesen werden, um in eine gute Position hinter dem Gegner zu gelangen. Jeder Treffer wird dem Schützen als Punkt zugeschrieben, und wer es zuerst schafft, den Kontrahenten 25 mal von der Fahrbahn zu fegen, hat gewonnen. Um die Jagd nicht übermäßig zu erschweren, prallen die abgefeuerten Projektile nicht etwa gegen das erste Hindernis, sondern folgen für einen bestimmten Zeitraum zufällig den Wegen des Labyrinths. Hat das Geschoß seine maximale Reichweite überschritten, verpufft es wirkungslos. Slotcars kann optional alleine oder zu zweit in verschiedenen Schnelligkeitsstufen gespielt werden. Die auf der Fish-Disk 254 enthaltene Version beinhaltet nur eine Strecke. Jedoch kann bei dem Programmautor eine Version bestellt werden, die zum einen einige zusätzliche Strecken für Slotcars beinhaltet und zum anderen noch drei weitere, völlig andere Spiele bietet.

Slotcars ist zwar ein recht einfaches Spiel, wurde aber mit viel Liebe zum Detail umgesetzt und bringt besonders zusammen mit einem weiteren Mitspieler viel Kurzweil.

Name: Slotcars
Enthalten auf: Fish 254
Vertrieb: siehe Anbieterliste



Ein Wettstreit um Macht und Reichtum

Wieder einmal geht es um Macht, Reichtum und Ehre. Königreiche müssen aufgebaut und verwaltet werden. Aber neben Wäldern, Wiesen, Bergen und Burgen macht ein Volk ein Königreich aus, und so gilt es, dieses bei Laune zu halten und das rechte Maß an Härte zu finden. In diesem Spiel können bis zu vier Spieler gegeneinander antreten und ihr wirtschaftliches und strategisches Geschick auf die Probe stellen.

Conquest of Illyria

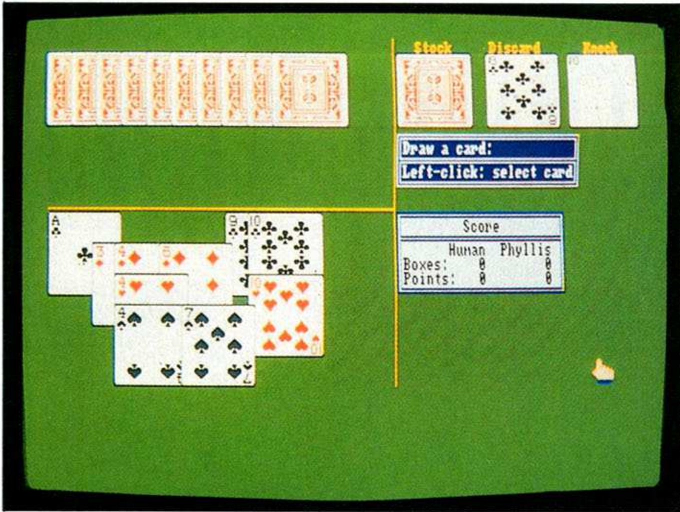
Gespielt wird auf einer 15 x 15 Felder großen Spielmatrix, die die Spielwelt darstellt. Jeder teilnehmende Spieler beginnt in einer der vier Ecken dieser Welt seine Eroberungszüge. Nach und nach müssen an die Heimat angrenzende Felder erobert und bewirtschaftet werden. Denn nur so kann jeder Herrscher die Ernährung und damit auch die Gefügigkeit seines Volkes sicherstellen. Je größer die Ländereien und je effektiver die Landwirtschaft, desto besser sind die überbleibenden Erträge, die zur Ausstattung der Armee benutzt werden können. Je besser wiederum die Armee gestellt ist, desto leichter gehen ihr neue Eroberungen von der Hand.

Erobertes Gelände kann durch den Bau von Burgen und Mauern vor den eifersüchtigen Nachbarn geschützt werden. Einige Spielfelder beherbergen außer Wald und Feld auch Goldvorkommen, die der Besitzer des betreffenden Feldes abbauen und für sich nutzen kann.

Von seiner Struktur her erinnert dieses Spiel stark an Brettspiele. Dieser Charakter wird besonders durch die recht einfache grafische Gestaltung und den Umstand unterstrichen, daß eine Solopartie, allein gegen den Computer, nicht möglich ist. Wer allerdings Freunde hat und diese zu einer Partie einlädt, kann ein spannendes Spiel erleben. Lästig ist nur, daß es ob der umfangreichen Spieldaten im Programmablauf bisweilen zu Wartezeiten kommen kann, aber das läßt sich verschmerzen.

Name: Conquest of Illyria
Enthalten auf: MDA 21
Vertrieb: siehe Anbieterliste

(Jürgen Seibel/hs)



Ein Kartenspiel mit ausgefuchsten Spielalgorithmen



Agentenhatz im geteilten Berlin: Wo sind die geheimen Pläne?

Cribbage King/ Gin King

Für Freunde gezinkter und ungezinkter Karten bietet diese Simulation zweier Kartenspiele einen guten Sparring-Partner.

Cribbage ist für den anglo-amerikanischen Raum in etwa das, was Skat für die germanischen Gefilde ist – ein sehr populäres Kartenspiel. Ähnlich verhält es sich mit Rummy, das dem hier bekannten Rommee entspricht. Kennen Sie diese Spiele? Möchten Sie sie lernen? Dann könnte dieses Paket genau das richtige für Sie und Ihren Amiga sein...

Besonderer Wert wurde auf die Vielfalt der Gegner gelegt. Jedes Spiel bietet sieben Computergegner, die unterschiedliche Strategien verfolgen. Man kann nicht nur gegen jede dieser Personen antreten, man kann sie auch gegeneinander spielen oder sich während des Spiels von ihnen beraten lassen. Während des Spiels liefert der Rechner Statistiken, man kann auch ein Spiel zwischenspeichern oder nachspielen. Der didaktische Effekt steht im Vordergrund, wodurch die Gestaltung auch ziemlich nüchtern wirkt.

Wer das Kartenspiel bislang nur aus Interesse an der Enthüllung von Damen im Sinne von Strip Poker betrieb, wird von diesem Programm sicher enttäuscht werden. Für den interessierten Kartenspieler, der entweder ein neues Spiel von der Pike auf lernen oder seine Fähigkeiten in einem bereits bekannten Spiel verbessern möchte, bietet dieses Programm eine interessante Möglichkeit zum Training.

(Michael Anton/hs)

AMIGA DOS Blitzlicht

Name: Cribbage King / Gin King
Hersteller: Software Toolworks
Vertrieb: Fachhandel
Preis: voraussichtlich 100 DM

Positiv:
- vielseitige Gegner
- "Lernprogramm"
- anwenderfreundlicher Kopierschutz
- Harddisk-Installation

Negativ:
- nüchterne Grafik
- wenig Sound



The Third Courier

Geheimpläne, Verräter, Wanzen, High-Tech: Der Dritte Kurier ist ein Spionagethriller in Reinkultur.

Drei Kuriere sollen die Einzelteile eines geheimen Verteidigungsplans der Nato ins Hauptquartier befördern. Leider entpuppt sich einer davon als Überläufer, der erst seine Kollegen umbringt und dann die Pläne einem östlichen Geheimdienst anbietet. Ein Zustandekommen dieses Geschäfts muß natürlich verhindert werden und ruft den Spieler in Gestalt eines Geheimagenten auf den Plan.

Im Spannungsfeld zwischen (dem damals noch geteilten) West- und Ostberlin müssen innerhalb von sieben Tagen die drei Teile aufgespürt und sichergestellt werden. Dabei steht dem Spieler ein mit allen erdenklichen Errungenschaften der Technik vollgestopftes Apartment zur Verfügung, zusätzliche Ausrüstungsgegenstände sind auf dem schwarzen Markt oder in der Berliner Zweigstelle der Organisation zu erwerben. Dort kann man sich auch, falls nötig, kurieren lassen. Mit U-Bahn, Taxi oder zu Fuß flitzt man in der Stadt herum, immer auf der Suche nach Informanten und Informationen.

Zu Beginn bastelt man sich einen Agenten nach eigenem Geschmack zusammen, dessen Eigenschaften sich während des Spiels immer weiter entwickeln. Auch die Personen, auf die man im Laufe des extrem komplexen Programms trifft, reagieren immer wieder anders und sind oft für Überraschungen gut.

Wenn der Sound auch auf Sparflamme kocht – die Grafik ist sehr sauber und funktionell gemacht (die Charaktere werden zum Teil sogar animiert dargestellt). Nach einem etwas langwierigen Einstieg läßt das menügesteuerte Programm den Spieler trotz einer gewissen politischen Realitätsferne nicht mehr los!

(A. Hink/hs)

AMIGA DOS Blitzlicht

Name: The Third Courier
Hersteller: Accolade
Quelle: Fachhandel
Preis: 84,95 DM

Positiv:
- komplexe Handlung
- bedienerfreundlich

Negativ:
- politisch nicht mehr aktuell
- Einstieg etwas mühsam





Warum haben es nur alle auf meinen Skalp abgesehen?

COLORADO

Arcade-Adventure oder Gemetzel, das ist die Frage bei diesem Franko-Western, wo das Blut in Strömen fließt.

Weit draußen im Wilden Westen ist das Leben eines Trappers alles andere als leicht. Überall lauern Indianer, die nichts besseres zu tun haben, als ein einsames Bleichgesicht in die ewigen Jagdgründe zu befördern. Da heißt es von Vorderlader, Tomahawk und Bowieknife ausgiebig Gebrauch machen. Friedliche Leute sind hier eher die Ausnahme. Da die meisten Indianer und Tiere sofort zum Angriff übergehen, sind Draufballern oder -hauen die einzigen Möglichkeiten, um ein allzufrühes Versiegen der Lebensenergie zu vermeiden. Doch Vorsicht mit dem Gewehr! Es steht nur eine begrenzte Menge an Pulver zur Verfügung. Ersatz dafür, aber auch Energie und Sprengstoff, ist bei einem fahrenden Händler im Tausch gegen Felle, Indianerschmuck oder Gold erhältlich.

Ärger gibt es nicht nur im Wald oder in den Bergen. Selbst bei einer friedlichen Paddeltour auf dem Colorado muß man sich mit schlecht gelaunten Rothäuten herumplagen. Vor lauter Ablenkung übersieht man dann schon mal mögliche Anlegestellen oder gar den Wasserfall...

Der Verlust des Lebens bringt auch den Schwachpunkt des Programms zum Vorschein. Der Fluß dieses eigentlich sehr hübsch gemachten Spiels leidet nämlich ziemlich unter der automatischen Save-Funktion, die einfach nicht flexibel genug ist. Die Steuerung per Joystick oder Tastatur allerdings ist exakt, wenn auch gewöhnungsbedürftig. Auch die Anwahl der Gegenstände im Inventory mittels Funktionstasten ist gut durchdacht. Sound und Grafik lassen kaum Wünsche offen.

(A. Hink/hs)

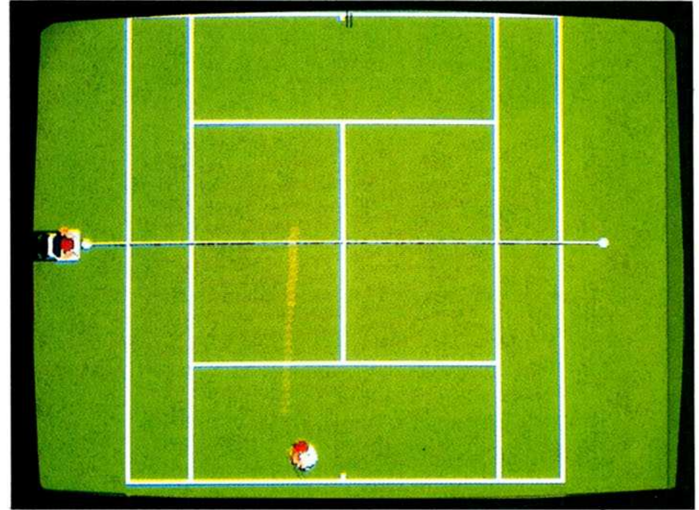
AMIGA DOS Blitzlicht

Name: Colorado
Hersteller: Silmarils
Quelle: Fachhandel
Preis: 84,95 DM

Positiv:
- Steuerung gut

Negativ:
- Spielstory naja
- "böser Indianer" als Feindbild
- Speichererweiterung muß abgeschaltet werden





Viele Schlagvarianten erfreuen den Tenniscrack

Tie Break

Sportspiele, im besonderen die Tennissimulationen, erleben in letzter Zeit einen Boom. Tie Break ist eine solche Tennissimulation, die gerade, weil sie einfach gehalten wurde, großen Spielspaß aufkommen läßt.

Die grafische Darstellung des Spielfeldes und der Figuren ist zwar nicht gerade schön, dafür aber sehr übersichtlich. Die Spieler werden aus der Vogelperspektive gezeigt. Der Ball wird, je nach Flughöhe, in verschiedenen Größen dargestellt.

Die menschlichen Mitspieler brauchen sich nur um das rechtzeitige Schlagen des Balles zu kümmern, das Laufen übernimmt der Computer für sie. Da keine Laufbewegungen nötig sind, können mit dem Joystick eine Menge verschiedener Schläge ausgeführt werden.

Bei Tie Break kann man die großen Tennisturniere selbst durchspielen, wobei auch festgelegt werden kann, wie viele Spieler an einem Turnier teilnehmen und wie viele davon der Computer übernehmen soll. Auf diese Weise ist es möglich, den Computer gegen sich selbst spielen zu lassen, um so Erfahrungen zu sammeln. Die Namen der Spieler kann man selbst festlegen oder auf die witzigen Vorgaben des Computers zurückgreifen, der bekannte Tennisgrößen in Namen und Bild parodiert.

Tie Break hat keine aufwendigen technischen Extras, sondern es wurde um der Spielbarkeit willen einfach gehalten. Die Möglichkeit, mit vielen Leuten gleichzeitig ein Turnier zu bestreiten und gegeneinander oder auch mal gegen den Computer mit den gewagtesten Schlägen zu kämpfen, läßt in jeder Spielrunde Stimmung aufkommen.

(Robert Marz/hs)

AMIGA DOS Blitzlicht

Name: Tie Break
Hersteller: Starbyte
Quelle: Fachhandel
Preis: 89,95 DM

Positiv:
- über Adapter bis zu vier Spieler gleichzeitig
- hoher Spielspaß
- übersichtliche Grafik

Negativ:
- mäßiger Sound



AMIGA DOS

Endlich keine Listings mehr abtippen!

Nicht bei allen Programmen ist es mit drei Zeilen getan – gute Routinen und praktische Funktionen brauchen ihren Platz. Und bisweilen lassen sich auch lange Datenblöcke nicht vermeiden, ganz zu schweigen von Hexdumps und Assemblerlistings.

Schonen Sie Ihre Augen und schlagen Sie sich nicht die Nacht mit Abtippen um die Ohren. – Auf der Databox zum Amiga DOS-Heft finden Sie alle Listings als ASCII-File: passend für jeden Texteditor, den Amiga-BASIC-Interpreter, Makro-Assembler oder einen Compiler für C und Modula-2.

Alle Programme sofort nutzen

Da ist er nun endlich – der Trick oder das Programm, auf das Sie schon so lange gewartet haben! Zu allem Unglück ist das Listing aber in Modula-2 oder C, jedenfalls in einer Sprache, zu der Sie keinen Compiler haben, um ein lauffähiges Programm herzustellen.

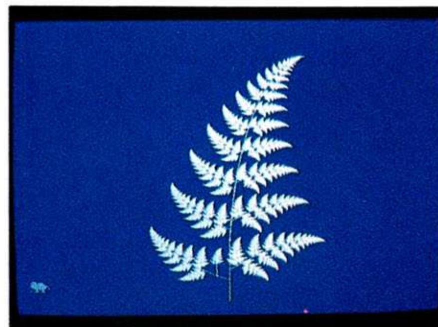
Auch in diesem Fall hilft Ihnen die Databox von Amiga DOS aus der Patsche: Neben den Quelltexten im ASCII-Format finden Sie jeweils auch das fertige, lauffähige Programm. Sie brauchen es nur von der Databox-Diskette aus zu starten.

DATA

Keinen Ärger mehr mit Tippfehlern

Wer kennt das nicht, wenn das Programm nach dem Eintippen nicht läuft oder der Rechner gar abstürzt. Besonders gemein sind auch Fehler, die erst nach Wochen bei einer bisher nicht gebrauchten Funktion zu Tage treten, oder wenn der Druckfehlerteufel am Werke war.

Zermartern Sie sich nicht den Kopf, bis Sie die falsche Zahl im Datafeld gefunden haben. – Alle Dateien auf der Databox zur Amiga DOS sind vom Autor und der Redaktion auf Fehlerfreiheit geprüft und im dazugehörigen System "probegelaufen".



Kopieren geht über studieren, ein etwas abgewandelter Spruch, der auf das Programm »Cli-Copy« zutrifft



Mit Turtlegrafik sollte man sich schon etwas näher beschäftigen. Eine reizvolle Art zu programmieren

Alle Listings und Programme auf Diskette – Computer einschalten – Diskette einlegen – los geht's

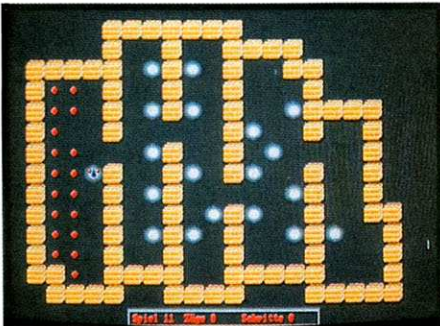


BOX

Inhalt der Databox AMIGA DOS 8/90

Basic	C	Modula-2	Assembler	Lauffähiges Programm	Inhalt der Databox AMIGA DOS 8/90
X				X	Die Sokoban-Umsetzung »Hard-Work« kann zu einer schweißtreibenden Angelegenheit werden
	X			X	Das Umbenennen von Devices ist mit »RenDev« kein Problem mehr
	X			X	Mit »CPointer« sagt uns der Mauszeiger, was die Uhr geschlagen hat
	X		X	X	Turtlegrafik ist durch Logo bekannt geworden; wir zeigen Ihnen einige interessante Beispiele
	X			X	Aufbau und Nutzen von Amiga-Libraries – funktionsfähiger Requester selbst programmiert
		X		X	Die ersten Befehle und Module präsentiert unser Modula-Kurs
			X	X	Wie programmiert man sein eigenes Kopierprogramm? Diese Frage beantwortet »Cli-Copy«
			X	X	**Und zusätzlich auf der Diskette:** Passend zum Titelthema "Musik" kommt unser Musikeditor »Speedy«





Harte Arbeit wird im wahrsten Sinne des Wortes mit »Hard-Work« geleistet

24, – DM

Wenn Sie über den DMV-Bestellservice bestellen, gilt folgendes:

Inland:		**Ausland:**	
Einzelpreis	24, – DM	Einzelpreis	24, – DM
zzgl. Versandkosten	4, – DM	zzgl. Versandkosten	6, – DM
Endpreis	**28, – DM**	**Endpreis**	**30, – DM**

Zahlungsweise:
Am einfachsten per Vorkasse (Verrechnungsscheck) oder als Nachnahme zuzüglich der Nachnahmegebühr. (Bei Lieferungen in das Ausland ist Nachnahme nicht möglich.)

Bitte benutzen Sie die Bestellkarte.

DMV-Verlag · Postfach 250 · 3440 Eschwege





Ölkrieg in den 90ern – unterwegs mit einem Zerstörer



Die beiden Ritter liegen im hartem Wettstreit um das Tal

U.S.S. JOHN YOUNG

Im Jahre 1997 hat sich die Situation auf der Erde stark verändert. Im Zuge der Rohstoffknappheit sind die arabischen Staaten zu Weltmächten geworden; die gesamte Nordhalbkugel ist von ihnen abhängig. Mit Ihrem privaten Zerstörer machen Sie sich auf, die Macht der Araber zu brechen.

Sie können zwischen vier Einsatzgebieten, die unter arabischer Herrschaft stehen, auswählen. Egal, für welche Mission Sie sich entscheiden, ob "Convoy", "Platform" oder "Battlefleet Attack", oberstes Ziel ist es immer, dem Feind soviel Schaden wie nur möglich zuzufügen.

Als Kommandant müssen Sie den Einsatzplan festlegen und die technischen Funktionen des Schiffs überwachen. Dazu können Sie von der Brücke aus in alle Stationen des Schiffs schalten.

Haben Sie ein gegnerisches Schiff oder Flugzeug zerstört oder eine Ölplattform vernichtet, schwärmt Ihre Crew automatisch aus und sucht nach verwertbaren Gegenständen. Diese können Sie in einem der beiden Ihnen freundlich gesonnenen Häfen, die es in jedem Gebiet gibt, verkaufen. Für das Geld können Sie Ihr Schiff dann neu bewaffnen und auftanken.

Technisch ist das Spiel gut realisiert, aber trotzdem will kein richtiger Spielspaß aufkommen. Oft geschieht es, daß das Schiff zerstört wird und der allzu bekannte Schriftzug "Game Over" auf dem Bildschirm erscheint, ohne daß man genau weiß, warum. Man hat keinerlei Möglichkeiten, zum Schluß die Todesursache noch abzufragen. Ein weiteres Manko sind die extrem langen Ladezeiten von Disketten. Glücklicherweise kann man das Spiel auf Festplatte installieren.

(Robert Marz/hs)

AMIGA DOS Blitzlicht

Name: U.S.S. John Young
Hersteller: Micropartner
Quelle: Fachhandel
Preis: 69,95 DM

Positiv:
- passable Grafiken
- auf Festplatte installierbar

Negativ
- kriegerische Handlung
- lange Disketten-Ladezeiten



Ritter

Ein Tal wird von zwei Rittern beansprucht, die ihre Burgen auf den gegenüberliegenden Bergen erbaut haben. Derjenige, der das Tal besser bewirtschaftet, wird zum Schluß die Mittel haben, die Burg des Gegners zu zerstören.

Am Anfang ist das gesamte Tal noch bewaldet, und jeder Burgherr hat eine Crew aus zwei Rittern, die das Tal kultivieren. Jedes der Felder, in die das Tal unterteilt ist, besitzt nämlich eine Wertigkeit, nach der die Punkte der Spieler zusammengerechnet werden. Zieht nun der Spieler mit seinen Rittern auf ein Feld, so wird dessen Wertigkeit erhöht. Im Klartext bedeutet das, daß der Wald gerodet, aus dem gerodeten Stück eine Wiese gemacht, darauf eine Holzhütte gebaut wird und so weiter. Zieht ein Ritter auf ein Feld, das ihm gehört, so schwärmt seine Gruppe auf Nachbarfelder aus und erhöht diese ebenfalls. Ist eine Fläche aus 3x3 Feldern kultiviert worden, kann dort ein Kastell errichtet werden, das Steuern zahlt und vom Gegner nicht einfach besetzt werden kann (wie das bei den einfachen Feldern der Fall ist).

Hat man sich zur gegnerischen Burg vorgekämpft und genügend Geld beisammen, kann man die Burg mit einem Katapult und sechs Steinen angreifen. Ist die Burg mit den sechs Steinen noch nicht vollständig zerstört, muß man in der nächsten Runde neu bezahlen. Ritter fehlt der richtige Pep. Für sein Geld bekommt der Spieler einfach zu wenig geboten: Ein wenig Handel oder ähnliches hätten die Programmierer ruhig mit einfließen lassen können. So steht das Spiel gerade am unteren Rand der Mittelmäßigkeit.

(Robert Marz/hs)

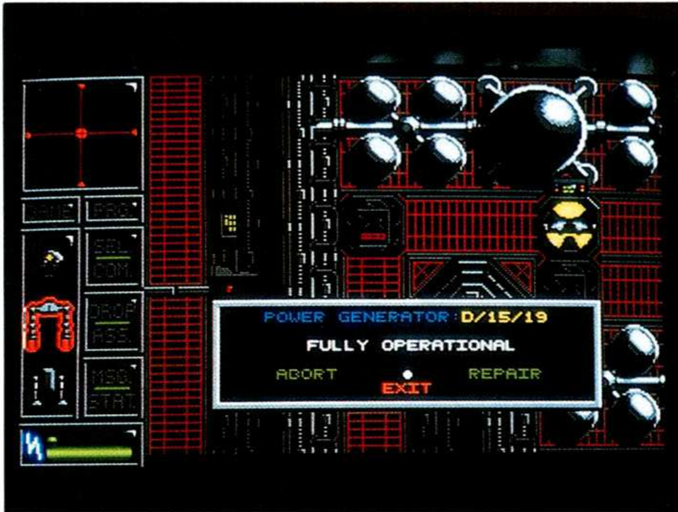
AMIGA DOS Blitzlicht

Name: Ritter
Hersteller: Ariola Soft
Quelle: Fachhandel
Preis: 69,95 DM

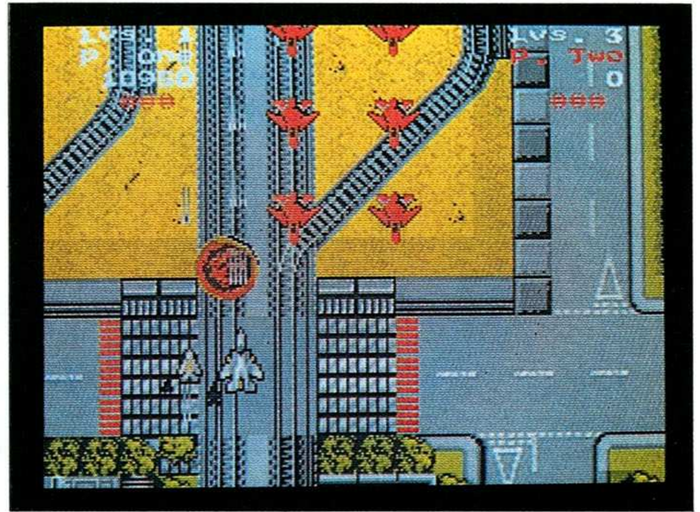
Positiv:
- deutsche Anleitung
- zwei Spieler oder Computergegner
- Spielstände speicherbar

Negativ:
- wenig Abwechslung
- fast kein Sound
- Computergegner in allen Spielstärken sehr schwach





Eine Raumstation voller Aliens und Abenteuer



Düsenjägerhatz, trotzdem ist Sonic Boom nicht gerade schnell…

First Contact

Die Erde befindet sich im Krieg mit einem übermächtigen, aber noch weitgehend unbekannten Gegner.

Der einzige Joker in dieser kosmischen Auseinandersetzung ist das Funkrelaisnetz der Erde, das bisher der Aufmerksamkeit der Aliens entgangen ist. Doch irgendwann ist es soweit, und die Außerirdischen entdecken und besetzen eine der Stationen. An Bord der Station befindet sich jedoch ein voll funktionierender Wartungs-Roboter, der aus drei Teilen besteht, die auch unabhängig voneinander operieren können. Sie, lieber Spieler, sind auserkoren worden, den Roboter über das Funknetz zu steuern und auf diese Weise die Station zu retten.

Die Station besteht aus vier Etagen, die jeweils aus der Vogelperspektive gezeigt werden. Jedes der vier Decks hat eine eigene Aufgabe. So ist Deck D zum Beispiel für die Energieversorgung der anderen Decks zuständig. Die Aliens demontieren und demolieren die Station. Fällt eines der Decks vollständig aus, wird die Verbindung zum Roboter unterbrochen, und die Station ist verloren.

In Spinden finden sich Gegenstände, die zu hilfreichen Energiesperren umgearbeitet werden können. Ist die richtige Code-Karte gefunden, kann man damit sogar Türen abschließen und die Aliens so aus bestimmten Sektoren fernhalten. First Contact verspricht durch die interessante Story und Aufmachung vor allem den Knobelfreunden viel Spaß, aber auch die Ballerfans kommen nicht zu kurz, denn die Aliens schießen zurück.

(Robert Marz/hs)

AMIGA DOS Blitzlicht

Name: First Contact
Hersteller: Rainbird
Quelle: Fachhandel
Preis: 89,– DM

Positiv:
- ausführliche, deutsche Anleitung
- Spielstände speicherbar
- Roboter vorprogrammierbar
- viele Parameter veränderbar

Negativ:
- mäßiger Sound



Sonic Boom

Ballerspiele, da scheiden sich die Geister. Was dem einen entspannende Unterhaltung verheißt, ist dem anderen zu blutrünstig. Trotzdem ist gerade dieses Genre immer wieder für Bestseller gut!

Schnelligkeit und Originalität sind hier gefragt. Doch ist bei weitem nicht alles empfehlenswert, was sich in diesem Rahmen auf dem Markt tummelt. Für jedes Superspiel à la R-Type tauchen ein dutzend Epigonen auf, die sich an Spielkonzept, Grafik und Aufmachung orientieren.

Sonic Boom ist ein Ballerspiel, in dem der Spieler die Kontrolle über einen Überschalljagdflieger übernimmt und damit diverse Ziele zerstören muß. Da gilt es Bohrinseln, Flugzeugträger und allerlei anderes zu beschießen. Das Spielfeld wird aus der Vogelperspektive dargestellt und scrollt vertikal unter dem Flugzeug dahin, während von oben immer neue Gegner auftauchen. Ab und an kutschiert mal ein Panzer durchs Gelände und schießt wahllos in die Szenerie. Formationen roter Flugzeuge beherbergen als Extras kleine Zusatzjets, die sich mit dem eigenen Jäger verbünden und so dessen Feuerkraft erhöhen. Das Scrolling läuft mit der Zähigkeit eines trockenen Kaugummis ab und ruckelt zudem noch ein wenig. Die Grafik ist hausbacken und wenig variationsreich; Sound und Musik sind nervig und nicht gerade neu. Das einzige was an diesem Produkt überdurchschnittlich ist, ist der Preis, und der sorgt dafür, daß die Wertung weit unter dem Durchschnitt liegt. Sonic Boom ist eigentlich nur ärgerlich, insbesondere für diejenigen, die dieses Produkt kaufen.

(hs)

AMIGA DOS Blitzlicht

Name: Sonic Boom
Hersteller: Activision
Vertrieb: Fachhandel
Preis: 84,95 DM

Positiv:
- stabile Verpackung

Negativ:
- zu teuer
- zu langsam
- zu wenige Levels
- mäßige Soundeffekte



Die meisten der Turmbauer bekamen es bei dem Anblick der Roboter mit der Angst zu tun und flüchteten.
Diejenigen aber, die dablieben, lernten viel von ihren neuen Freunden und trieben den Turmbau voran. Die Roboter, die aussahen wie riesige Spinnen, mußten nämlich so hoch wie möglich kommen, um mit ihrem Raumschiff wieder in Kontakt treten zu können. Die Menschen wollten aus den bekannten Gründen in die Höhe. Als die Menschen erkannten, daß ihre Freunde von Heimweh geplagt waren und so schnell wie möglich nach Hause wollten, entbrannte in ihnen eine rasende Eifersucht; sie wollten sie für sich behalten. So machten sich einige böse Menschen daran, den Turm mit Fallen zu spicken, um die Roboter am Fortkommen zu hindern. Die Menschen, die nichts mit dem Untergang der Roboter zu tun haben wollten, flüchteten, so daß nur noch böse Menschen übrigblieben.
Die drei Spinnen aber, von denen jede eine herausragende Eigenschaft hat, machen sich auf den gefahrvollen Weg nach oben und bauen trotz aller Widrigkeiten an dem Turm weiter. Sie, lieber Spieler, sollten ihnen zu Hilfe kommen, um das Unrecht der bösen Menschen an den Spinnen wiedergutzumachen.
Die drei Spinnen müssen die einzelnen Turmabschnitte fertigstellen. Dazu müssen Sie entweder eine gewisse Anzahl von Klondikes einsammeln, den Turm von exotischem Abfall säubern oder beides.
Dazu stehen Ihnen die drei Spinnen zur Verfügung, Zapper, Pusher und Grabber.
Zapper besitzt eine Kanone, mit der er Gegenstände, aber auch die eigenen Spinnen abschießen kann. Pusher hat anstatt der Kanone einen Rückstoßstrahl, der Gegenstände verschieben kann. Grabber schließlich kann Gegenstände anziehen, und er ist der einzige, der die Klondikes aufsammeln kann.
Am Anfang des Spiels betreten die Spinnen den ersten Abschnitt des Turmes. Sie erfahren den Namen des Abschnitts, der meistens Aufschluß über die Lösungsmöglichkeit gibt, und die Aufgabe, also wieviele Klondikes eingesammelt und wieviele Gegner abgeschossen werden sollen. Außerdem erfährt der



Die Spinnenroboter müssen den Turm zu Babel fertigstellen, um nach Hause funken zu können

Tower of Babel

Das größte Projekt der vorchristlichen Menschheit war der Turmbau zu Babel. Er erlangte solchen Ruhm, daß sogar die Zantorianer, ein außerirdisches, freundliches Volk, davon erfuhren. Sie überließen den Erdlingen drei Roboter, die ihnen zur Hand gehen sollten.

Spieler, welche der Spinnen zur Verfügung steht, denn es gibt Abschnitte, die nur mit einer oder zwei Spinnen gelöst werden müssen.
Bevor der Countdown einsetzt und die einzelnen Gegenstände im Abschnitt sich zu bewegen beginnen, kann sich der Spieler in Ruhe umsehen. Er nimmt das Szenario wahr, als würde er durch die Kameraaugen der Spinnenrobots schauen. Diese können zoomen und sich nach rechts und links bewegen, so daß jeder Punkt des Turmes vor Beginn des eigentlichen Spieles genau unter die Lupe genommen werden kann. Tut einer der Roboter einen ersten Schritt oder schießt er, erwacht der Turm zum Leben. Würmer und Hüpfer fangen an sich zu bewegen, feststehende Strahler, Zieher und Greifer werden aktiv und Fahnen flattern im Wind. Man sollte schon einen genauen Plan haben, denn die Abschnitte lassen sich meist nur auf eine Weise lösen, und es ist eine ganze Menge Logik vonnöten, um es zu schaffen.
Es gibt eine Reihe von Gegenständen, die sich in einem Abschnitt befinden können. Da hüpft Kanonenfutter für Zapper durch die Gegend, das aber nur für ihn hilfreich ist und allen anderen im Weg herumhüpft. Da gibt es Steine, die in der Gegend herumliegen und nur aus dem Weg geschoben werden können, Prismen, die Schüsse umlenken, Glasblöcke, die sie reflektieren und vieles, vieles mehr.
Tower of Babel ist ein Strategie- und Logikspiel der Extraklasse. Eine ausgezeichnete Idee wurde technisch sehr gut umgesetzt und verspricht durch viele verschiedene Türme langen Spielspaß. Wem das noch nicht reicht, der kann sich mit dem beigelegten Level-Editor komfortabel eigene Türme basteln und sie seinen Freunden vorsetzen.
Bei Tower of Babel besteht absolute Suchtgefahr für alle, die gerne an schwierigen Aufgaben knobeln.

(Robert Marz/hs)



Daß diese Aufgabe weder leicht noch ungefährlich ist, dürfte einleuchten

AMIGA DOS Blitzlicht

Name: Tower of Babel
Hersteller: Microprose
Quelle: Fachhandel
Preis: 89,– DM

Positiv:
- Anleitung sehr ausführlich und komplett deutsch
- Level-Editor enthalten
- viele Parameter einstellbar
- Spinnen im voraus programmierbar

Negativ:
- Sound etwas mager





Krieg als Ballerspiel, mit P 47 als Kampfpilot unterwegs

P 47

Auch wenn die Rettung des Universums eine nette Aufgabe ist, verlegt P 47 den Schauplatz einer wilden Ballerjagd aus den Weltraumtiefen auf diverse historische Kriegsschauplätze.

P 47, so lautet die Typenbezeichnung eines während des zweiten Weltkriegs verbreiteten Kampfflugzeugs, mit dem der Spieler in diesem Programm zu Einsätzen an diversen Schauplätzen startet. Es gilt, sich die Formationen feindlicher Kampfflugzeuge vom Leibe zu halten und die Geschützstellungen auf dem Boden zu meiden. Natürlich wird aus allen Rohren auf die P 47 gefeuert, und gäbe es da nicht noch einige Extras, die die Bewaffnung des Jägers ergänzen, wäre P 47 schlicht und ergreifend zu schwer. Aber Granatwerfer, zusätzliche MGs und Bomben für Bodenziele verschaffen ein wenig Luft zum Manövrieren. Am Ende jeder Spielzone lauert ein besonders wehrhafter und fieser Wächter, der niedergerungen werden muß. Der Anfang ist leicht, und der Kanonenzug, der in Zone eins das Levelende bewacht, ist mit ein paar Bomben schnell erledigt.

Ballerspiele sind beliebt. Ob das auch über P 47 gesagt werden kann, ist nur schlecht zu beantworten. Zwiespältig ist die Mischung aus guter Grafik, passabler Musik, durchschnittlichem Programm und passablem Spielgeschehen schon. Da fliegen Geschosse mitten durch Berge, Kollisionen erfolgen ohne Hindernisse, und nachdem die Maschine auf dem Monitor erscheint, ist sie zwar unverwundbar (angenehm, wenn gerade viele Gegner auf dem Bildschirm sind), kann aber auch keine Extrawaffen aufnehmen (eher unangenehm). Den Oscar für das "Ballerspiel des Jahres" bekommt P 47 definitiv nicht.

(hs)

AMIGA DOS Blitzlicht

Name: P 47
Hersteller: Firebird
Vertrieb: Fachhandel
Preis: 89,– DM

Positiv:
- gute Grafik
- netter Sound

Negativ:
- spielerische Mängel
- stellenweise zu langsam





Spiel geschwind, 5 gewinnt, doch meist schafft es das Himmelskind

Antago

Kennen Sie "Vier gewinnt"? Antago ist eine höllisch kniffelige und himmlisch spielbare Variante dieses Klassikers.

Da gibt es immer noch Leute, die behaupten, der Himmel sei friedlich. Die haben wohl Antago noch nicht in die Hände bekommen. Hier liefert man sich knallharte Duelle, um die Seele eines Verblichenen zu gewinnen. Der Spieler schlüpft dabei in die Rolle des Teufels. Als Gegner tritt, wie könnte es auch anders sein, ein Engel zum Duell an. Das Spiel gestaltet sich im "Fünfgewinnt"-Stil. Ziel ist es, in einem 5x5 Felder großem Spielfeld fünf Spielsteine in eine Reihe zu bekommen. Dabei schiebt der Teufel Minen auf das Spielfeld und der Engel Wölkchen. Der Engel beginnt in jedem Fall das Spiel und legt einiges an Spielstärke an den Tag. Wolken und Minen können auch verschoben werden. So kommt es immer wieder zu kniffeligen Situationen, in denen es erforderlich ist, ein paar Schritte vorauszudenken, um nicht vom Engelchen überlistet zu werden.

Die hübsch gezeichneten und sauber programmierten Animationen können zwar begeistern, verlieren aber bald ihren anfänglichen Reiz. Insbesondere, wenn eine Figur um zwei Seiten des Spielfelds herumwandern muß, um den Spielstein an die gewünschte Stelle zu setzen, ist Geduld angesagt. Die Spielstärke des Programms kann in einem Menübildschirm eingestellt werden, so daß die Leistungen des Computerspielers nach Wunsch regulierbar sind.

Insgesamt stellt sich dieser Versuch, einen Klassiker im neuen Gewand zu präsentieren, positiv dar und ist für Liebhaber einfacher Denkspiele eine empfehlenswerte Sache.

(T. Siebert/hs)

AMIGA DOS Blitzlicht

Name: Antago
Hersteller: Art of Dreams
Vertrieb: Fachhandel
Preis: 59,95 DM

Positiv:
- gute Grafik
- bewährtes Spielkonzept

Negativ:
- bisweilen Wartezeiten





Passionate Patti, Larrys weiblicher Gegenpol in diesem Spiel

Leisure Suit Larry III:

Passionate Patti In Pursuit Of The Pulsating Pectorals

ER ist wieder da: Larry Laffer, der bemitleidenswerte Triebtäter im weißen Polyesteranzug (pflegeleicht und gefühlsecht). Auch in dieser Folge ist er wieder auf der Suche nach dem weiblichen Geschlecht...

Daß sich Larry bei allen seinen Liebschaften letztendlich in die Nesseln setzt, dürfte seinen Fans und ständigen Begleitern langsam bekannt sein. So verwundert es auch nicht, daß Larry zu Beginn des Adventures "wie üblich" dasteht: keine Frau, kein Geld und nur "das eine" im Kopf... Die Inselschönheit der letzten Folge hat ihn aus dem gemeinsamen Eigenheim geworfen, seinen Job in der aufblühenden Tourismusbranche von Noontonyte hat er mangels Talent und Engagement verloren. Also besinnt er sich auf seine "natürlichen Talente", kramt den Polyesteranzug wieder hervor und beginnt seinen Streifzug über die Insel.

Mit kleinen Tricks und mehr oder weniger großzügigen Geschenken kann Larry tatsächlich bei einigen Damen landen, die Ergebnisse sind jedoch nicht immer sehr berauschend. Fast immer kommt etwas dazwischen, wenn der Ärmste sich ausleben will: Ob bei der Strandschönheit, einer Tänzerin, der etwas seltsamen Anwältin oder der Aerobic-Tänzerin – am Ende der Affären steckt Larry in einer unangenehmen oder peinlichen Situation. Doch plötzlich kommt Passionate Patti ins Spiel, gleich Larry von unbändiger Lebensfreude und mit entsprechenden Problemen bestens vertraut. Die beiden sind wie füreinander geschaffen, und diesmal scheint es wirklich die große Liebe zu sein. Leider kommt es zu einem Mißverständnis, welches Larry dazu veranlaßt, seinem bisherigen Lebenswandel zu entsagen. Also verläßt er die Frau seiner Träume, um sich in Enthaltsamkeit zu üben.

Patti ist davon nicht sehr angetan und macht sich auf die Suche nach Larry. Hier kommt nun die große Wende: Während der Spieler bislang Larry hilfreich zur Seite stand, steuert er nun Patti durchs Adventure-Leben. Diese hat einen langen Weg vor sich, den sie durch den Einsatz ihrer Intelligenz und ihrer Kleidung bewältigt – er gleicht einem Striptease-Puzzle. Am Ende findet sie den Angebeteten natürlich, und es kommt zum Happy-End – obwohl das Finale nicht unbedingt überwältigend ist...

Die technische Gestaltung des Spiels überzeugt jedoch. Die Grafik ist zwar nicht ganz so Amiga-bunt (es ist halt doch "nur" eine Umsetzung vom PC), aber dennoch ansprechend und gut animiert. Angenehm fällt auch der Sound auf; fast zu jedem Bild wird eine andere Musik geliefert, die hörenswert ist. (Leider wurde wieder auf die Übernahme der beim PC vorhandenen MIDI-Unterstützung verzichtet, auch fehlen einige dort vorhandene Stücke in der Amiga-Version.) Der Schwierigkeitsgrad ist eher moderat, routinierte Sierra-Freaks dürften es fast als ein Kinderspiel empfinden. Garantiert nicht für Kinder ist die Thematik, die doch recht schlüpfrig ist. Waren die beiden ersten Teile noch relativ harmlos, sei es durch Zensurbalken oder mangels Gelegenheit, so geht es hier sehr offen-sichtlich und teilweise animiert "zur Sache". (Der "Schmutzgrad" wird nach einer anfänglichen "Reifeprüfung" in Form von fünf Fragen festgelegt.) Alles in allem ist Larry III trotz einiger Schwächen empfehlenswert, da er (wie üblich) mit vielen Gags, guten Animationen und sarkastischen Kommentaren aufwarten kann. (Und einer Kultfigur verzeiht man auch manchen Ausrutscher...)

Aber auch wenn um Larry und seine Abenteuer eine regelrechte Kultgemeinschaft entstanden ist, so zeigt das Spielkonzept für Spieler, die keine ausgesprochenen Larry-Fans sind, erste Ermüdungserscheinungen.

Eigentlich nur natürlich, dreht es sich doch in der ganzen Serie nur um das eine. So erklärt sich auch die von Teil zu Teil der Serie immer weiter forcierte Deutlichkeit, was das Spielgeschehen und dessen Darstellung betrifft.

Was bleibt, ist ein gut umgesetztes Grafikabenteuer, das in Kinderhänden nichts zu suchen hat.

(Michael Anton/hs)





Bars und Kneipen, hier fühlt Larry sich nach wie vor wohl

AMIGA DOS Blitzlicht

Name: Leisure Suit Larry III: Passionate Patti In Pursuit Of The Pulsating Pectorals
Hersteller: Sierra
Vertrieb: Fachhandel
Preis: 119,95 DM

Positiv:
- Figurenwechsel
- gute Animation
- guter Soundtrack
- freundlicher Kopierschutz

Negativ:
- zu einfach
- zu offen
- Festplatteninstallation läuft nicht immer

AMIGA DOS URTEIL GUT

GRAFIK	SOUND	MOTIVATION
7	8	7



Ein Maulwurf als Superheld – Monty Mole rettet die Welt!



Der tapfere Ritter als Schrumpfgermane

Impossamole

Monty der Maulwurf ist wieder da. Diese Figur erlebte während der großen Zeit der Acht-Bit-Computer ganze Serien von Computerspielen als Held. Nun ist der Star der 80er wieder da und besteht sein erstes Abenteuer in den 90ern.

Außerirdische haben Monty entführt und ihn mit Superfähigkeiten ausgestattet. Aus Monty wurde Impossamole, ein Held, der mit fast jeder Schwierigkeit fertig wird. Natürlich haben die Aliens Monty nicht ohne Grund zum "Supermundschmeiß" gemacht, er soll für sie die Kastanien aus dem Feuer holen und fünf extrafiese Wächter besiegen.

Hier dürfen Sie sich auf ein Jump'n'Run-Spiel freuen, das bestimmt nicht nur Kindern Spaß macht. Durch fünf verschiedene Landschaften muß Monty wandern. Dabei darf er den vielen Gefahren, die auf ihn lauern, nie zu nahe zu kommen. Der Spieler kann von Anfang an unter vier verschiedenen Schauplätzen wählen. Den fünften kann er jedoch erst aufsuchen, wenn die anderen vier gemeistert wurden. So geht es durch unterirdische Minen, vorbei an orientalischen Schauplätzen, durch Amazonas und Antarktis bis zur letzen Konfrontation im Bermuda-Dreieck. Unterwegs kann Monty immer wieder Gegenstände finden, die sein Leben erleichtern: Dosen voll Lebensmittel, Granatwerfer und Schätze.

Obwohl es um das Genre der Jump'n'Run-Spiele etwas ruhiger geworden ist, erfreuen sich Programme dieser Machart auch auf dem Amiga eines regen Zuspruchs. Impossamole sei allen Freunden des Genres ans Herz gelegt, denn einerseits spielt es sich gut, und andererseits ist es verhältnismäßig preiswert.

(hs)

AMIGA DOS Blitzlicht

Name: Impossamole
Hersteller: Gremlin
Vertrieb: Fachhandel
Preis: 64,95 DM

Positiv:
- passable Grafik
- gute Musik

Negativ:
- stellenweise sehr schwer



Fred

Was tun, wenn Sie als mächtiger Ritter in ein kleines, schwaches Kerlchen verwandelt werden? Na klar, Sie suchen den Zauberer auf, der Ihnen das angetan hat.

Fred, das sind Sie, befindet sich nach seiner Verwandlung in einem finsteren Wald, dem Reich der Zwerge. Um den Zauberer zu finden, muß er das Reich durchqueren. Als er noch groß war, hätte ihm das nichts ausgemacht, aber für den kleinen Dreikäsehoch, der mit nichts außer seinem Schwert und ein paar Dolchen bewaffnet ist, sieht die Sache schon ganz anders aus.

Fred bewegt sich durch die horizontal scrollende Landschaft und kann dabei Messer werfen oder mit seinem Schwert schlagen. Bei seinem Lauf bewegt er sich aber nicht nur nach links oder rechts, sondern er kann auf mehreren, parallelen Wegen, zum Beispiel vor einem Baum oder hinter einem Baum, entlang laufen. Seine Gegner sind in den einzelnen Levels hauptsächlich Zwerge, aber auch Spinnen, Schlangen, Vögel und Hornissen. Die Zwerge gibt es in allen Variationen: Zwergenkinder, die Fred mit der Steinschleuder beschießen oder bierfässertragend und morgensternschwingend durch die Gegend laufen, Erwachsene mit Speeren und Pfeil und Bogen oder Wirte, die mit Metallkrügen schmeißen. Diese Reihe ließe sich noch sehr lange fortsetzen.

Fred zeichnet sich durch seinen moderaten Schwierigkeitsgrad, der keinen Frust aufkommen läßt, und die wunderschöne Grafik, die immer wieder zum Lachen anregt, aus und ist in jedem Falle Freunden diesen Genres heiß zu empfehlen.

(Robert Marz/hs)

AMIGA DOS Blitzlicht

Name: Sir Fred
Hersteller: Ubi-Soft
Quelle: Fachhandel
Preis: 89,– DM

Positiv:
- sehr gute Grafik
- nicht zu hoher Schwierigkeitsgrad
- viel Abwechslung
- diverse Extras

Negativ:
- Steuerung gewöhnungsbedürftig



Hero's Quest I: So you want to be a Hero...

Als ein Adventure mit Rollenspielelementen präsentiert sich das neue Epos aus dem Hause Sierra. Ein kleiner Mickerling muß sich zum gestandenen Abenteuerer entwickeln und dabei ein Tal aus den Klauen des Bösen befreien.

Eigentlich verlief das Leben in der Stadt Spielberg und den umgebenden Landen lange Zeit friedlich – die Diebe klauten, die Händler handelten und die Herrscher regierten. Mit anderen Worten: Stinklangweilig war es... Das änderte sich aber schlagartig, als sich der Baron von Spielberg mit Baba Yaga anlegte, einer zugereisten Hexe. Diese legte einen Fluch über das Land, und von da an war alles anders. Miese Monster wagten sich bis kurz vor die Hütten der Menschen; Orcs, Räuber und diverse andere Ungeziefer machten die Wälder unsicher.

Der Zauber der guten Fee Erana konnte zwar etwas Einhalt gebieten, aber mit Zaubern allein ist es nicht getan. Nach einer Prophezeiung muß erst ein Held von jenseits der Berge kommen und mit dem Bösen aufräumen, bevor wieder Frieden im Lande herrscht. Viele haben es schon versucht und sind gescheitert, so daß die Lage langsam hoffnungslos wird.

Da kommt eines Tages ein junger Abenteurer in diese Gefilde – beseelt von dem Wunsch, ein Held zu werden, aber doch noch recht grün hinter den Ohren und schwach auf der Brust. Dieses Greenhorn zu einem wackeren Recken zu machen, ist die Aufgabe des Spielers. Die ist zu Anfang gar nicht leicht, aber im Laufe der Zeit lernt man dazu und verbessert auch die Fähigkeiten der Figur. Die Erfolgsaussichten können auch durch Kauf von Waffen, Zaubertränken und -sprüchen verbessert werden. Das Geld für die Reisekasse kann man sich durch Übernahme verschiedener Aufträge beschaffen (so sind Sohn und Tochter des Barons zu finden), der geneigte Dieb kommt in einer Nachtschicht an Valuta, und auch die im Walde erlegten Monster haben oft Wertgegenstände bei sich. Durch Befragen der Anwohner erfährt man Wichtiges, ansonsten ist man auf Glück und Intuition angewiesen.

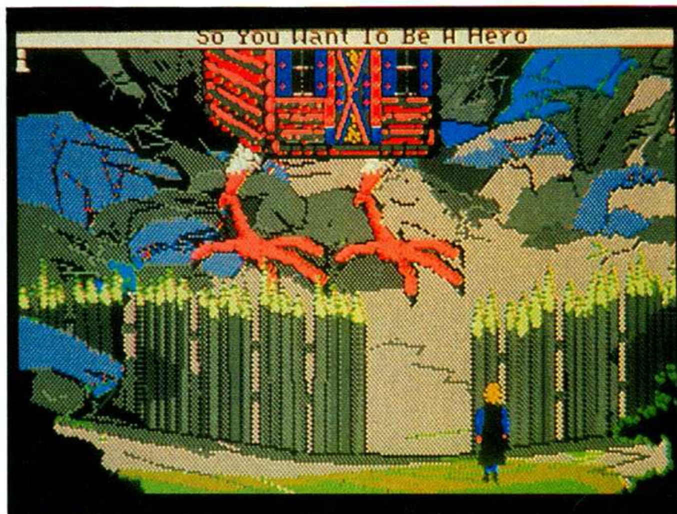
Grafisch ist Spielberg recht ansprechend gestaltet, auch der (sparsam verteilte) Sound kann sich hören lassen – neuerer Sierra-Standard eben... Das Rollenspiel kommt bei der Wahl der Figur zum Tragen: Drei Charaktere stehen zur Auswahl, ein Kämpfer, ein Dieb und ein Magier. Jeder von ihnen ist zu Anfang mit verschiedenen Fähigkeiten ausgestattet, die vor dem Start noch durch frei zu vergebende Zusatzpunkte variierbar sind. Im Laufe des Spiels verbessern sich diese Fähigkeiten: Intelligenz beim Lösen von Rätseln, Stärke im Kampf oder Magie durch Zaubern. Neben solchen globalen Skills können jedoch auch andere Fähigkeiten trainiert werden wie das Öffnen von Schlössern oder das Klettern.

Als Rollenspiel kann Hero's Quest nicht unbedingt empfohlen werden, als ein "etwas anderes" Adventure von Sierra kann es jedoch gut überzeugen. Daß man seine Figur erst aufbauen muß, mag zwar anfangs nerven, aber ist man erst einmal in der Welt von Spielberg gefangen, so macht es doch Spaß. Die Rätsel sind teilweise recht kernig, für die meisten gibt es mehrere Lösungen, die von den Eigenschaften und der Ausrüstung des Spielers abhängen. Wer schon jetzt seinen Helden gut aufbaut, erspart sich diese Arbeit in der bereits angekündigten Fortsetzung, denn dort kann der Charakter weiterverwendet werden.

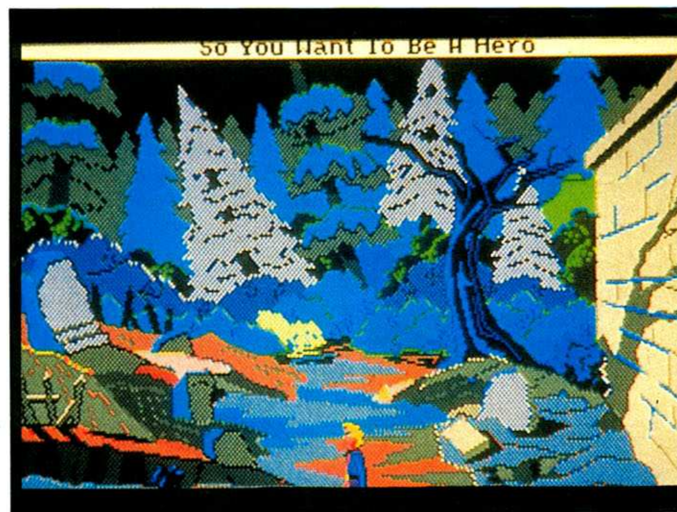
Hero's Quest hebt sich auf jeden Fall von den meisten anderen Sierra-Abenteuern ab. Die Machart ist zwar im wesentlichen die gleiche geblieben, jedoch hat der Spieler mehr Möglichkeiten, seinen Spielcharakter zu entwickeln. Knallharten Rollenspielern, für die Ultima oder Questron der Stein der Weisen bezüglich des Spielspaßes darstellt, sei jedoch gesagt, daß Hero's Quest weniger ihr Fall sein wird.

Und vielleicht noch ein kleiner Tip: Am besten hat sich ein Dieb bewährt. Läßt man diesen noch das Zaubern lernen und meuchelt fleißig Monster, so hat man am Ende einen Universalcharakter, der allen Lagen gewachsen ist. (Ein kleiner Gag am Rande: Mit einer solchen Figur kann man das Spiel mit 605 von 600 Punkten beenden!)

(Michael Anton/hs)



Wer Held werden will, muß sich bewähren, auch beim Monstermeucheln



Auch sonst gibt es in diesem Spiel viel zu erkunden

AMIGA DOS Blitzlicht

Name: Hero's Quest I
Hersteller: Sierra
Vertrieb: Fachhandel
Preis: 119,95 DM

Positiv:
- "neues" Prinzip
- anspruchsvolle Rätsel
- freundlicher Kopierschutz

Negativ:
- Kampfsequenzen teilweise "lahm"
- Festplatteninstallation läuft nicht immer

AMIGA DOS URTEIL GUT

GRAFIK	SOUND	MOTIVATION
8	6	8

Die Mischung macht bereits auf den ersten Blick neugierig. Spherical, Hard'n'Heavy, Circus Attractions und Grand Monster Slam sind Titel, die schon beim ersten Erscheinen für Aufsehen sorgten und nun zu einer preiswerten Programmsammlung zusammengestellt wurden.

Bei **Spherical** muß ein kleiner Zauberer einer Kugel durch ein Labyrinth zum Ausgang verhelfen, was durch herbeigezauberte und geschickt plazierte Steine geschieht. Die Kugel folgt den Gesetzen der Schwerkraft; der Weg, den sie dabei nimmt, führt jedoch nicht immer zum Ausgang. Zum Glück bleibt sie zu Beginn eines Bildes erstmal ruhig liegen, so daß sie zunächst "eingemauert" werden sollte. Dies ist nötig, da die Ruhezeit nur selten reicht, um den rechten Weg zu legen. Darüber hinaus gibt es in den Bildern auch Bonusgegenstände, die eingesammelt werden sollten. Über 100 Bilder warten darauf, gelöst zu werden. Passwörter und geheime Levels gehören ebenso zum Spiel wie Gegner, die bei der Arbeit behindern. Neue Perspektiven eröffnet auch ein Modus für zwei Spieler, der völlig neue Levels zeigt. Die Grafik ist putzig animiert, sechs verschiedene Musikstücke versüßen das Ableben der Figur. **Spherical** ist ein Spiel für alle Puzzle- und Strategiefreunde, das süchtig machen kann. Schon allein dieses Spiel lohnt die Anschaffung des Samplers!

Hard'n'Heavy ist ein Hüpf- und Sammelspiel, bei dem ein kleiner Roboter möglichst viele Kristalle einsammeln muß, was ihm durch umherspringende und -fliegende Gegner erschwert wird – Kollisionen sind tödlich. Insgesamt warten 25 Levels darauf, erforscht zu werden; Geheimlevels und versteckte Schatzhöhlen sind ebenfalls zu entdecken. Für Abwechslung sorgen auch vielfältige Optionen für zwei Spieler. Hat man sich erst einmal mit der etwas "weichen" Steuerung vertraut gemacht, warten viele Stunden Spielvergnügen. Das wird verstärkt durch die gute und abwechslungsreiche Musik sowie eine nett animierte Grafik – sieht man mal von dem etwas eintönigen Hintergrund ab. Erfolge werden in Highscore-Listen abgelegt, wobei sowohl Punktzahl als auch



Hard'n'Heavy, hier darf gehüpft werden

Milestones

Geschicklichkeit ist beim neuen Sampler von Rainbow Arts angesagt. Statt heftigem Joystick-Schwingen sind ruhige und bedachte Aktionen bei diesen vier Spielen etwas älteren Datums gefragt.

Fortschritte in den jeweiligen Levels gespeichert werden. Für Freunde dieser Art von Spielen kann **Hard'n'Heavy** nur empfohlen werden.

Den Zauber der Manege hautnah erleben kann man in **Circus Attractions**. Fünf verschiedene Zirkusnummern können gespielt werden: Trampolinspringen, Seiltanz, Jonglieren, Messerwerfen und Wippenspringen sind die Disziplinen, die für eine erfolgreiche Tournee beherrscht werden müssen. Bis zur Perfektion ist jedoch viel Übung angesagt, da einige Disziplinen etwas unübersichtlich sind. Leider fehlen dem Spiel etwas "Biß" und Abwechslung, so daß auch die gute Grafik und die stilechte Musik nur wenig zu einem längeren Spielspaß beitragen. Aber sicherlich finden sich auch für solche Spiele Liebhaber...

Wie Sport in Vergangenheit, Gegenwart und Zukunft aussieht, ist bereits zur Genüge simuliert. **Grand Monster Slam** wendet sich der Frage zu, wie Sport in einer Fantasy-Welt aussieht. Besagtes Slam-Turnier findet regelmäßig auf der Sechssonnenwelt statt und besteht überwiegend darin, kleine Pelzkugeln, Beloms genannt, zu treten – sei es, um sie auf die gegnerische Seite des Spielfelds oder in die Mäuler hungriger Faultons zu kicken. Ziel ist es, möglichst viele Runden gegen verschiedene Gegner zu überstehen. Grafik, Sound und besonders die Animationen sind hervorragend gelungen und reizen zu Lachmuskelkater. Leider wird das Spiel trotz allem mit der Zeit langweilig, da wenig Neues zum Vorschein kommt.

Schade, denn aus dieser Monster-Olympiade hätte man etwas mehr machen können.

Bedenkt man, daß man bis vor kurzem noch für jedes einzelne Spiel auf diesem Sampler dieselbe Summe investieren mußte, die heute vier Programme kosten, ist Milestones allemal ein Gewinn. Hinzu kommt, daß die präsentierte Auswahl Überdurchschnittliches zu bieten hat. Da kann man durchaus über die Schwächen des Samplers hinwegsehen, die sich auch hier finden lassen. Daß alle enthaltenen Programme im wesentlichen friedlich sind, unterstreicht die Attraktivität des Paketes zusätzlich.

Die Noten, die Sie in unserem Bewertungsschema finden, sind diesmal natürlich Durchschnittswerte, die sich auf alle enthaltenen Spiele beziehen.

(Michael Anton/hs)



Bei Spherical ist Magie im Spiel

AMIGA DOS Blitzlicht

Name: Milestones
Hersteller: ReLine
Vertrieb: Fachhandel
Preis: 79,95 DM

Positiv:
- günstiger Preis
- grafische und akustische "Leckerbissen"

Negativ:
- einige Spiele etwas "lasch"

AMIGA DOS URTEIL GUT

GRAFIK	SOUND	MOTIVATION
8	8	7



Abenteuer im Drachenland



Puzzlespiel mit Atomen

Theme Park Mystery

Einen Jahrmarkt erbt man nicht alle Tage; einen, auf dem es spukt und auch ansonsten nicht alles mit rechten Dingen zugeht, noch viel seltener.

Ihnen widerfährt ein solcher "Glücksfall". Doch wie sich recht bald zeigt, verbirgt sich Unheimliches in diesem Vergnügungspark, und die Besucher bleiben aus. Einzige Chance, den drohenden Bankrott abzuwenden, besteht darin, aus dem Park wieder ein funktionierendes Unternehmen zu machen. Dazu gilt es allerdings zunächst einmal, die boshaften kleinen Dämonen unschädlich zu machen, die sich in dem Jahrmarkt eingenistet haben. Die kleinen Unholde haben sich in den verschiedenen Attraktionen des Parks versteckt, und der Spieler muß diese nun eine nach der anderen aufsuchen und die winzigen Fieslinge einsammeln. Da gibt es das märchenhafte Drachenland, Traumland und Zukunftsland. Jede dieser Attraktionen stellt ein Spiel im Spiel dar, die der Spieler nach und nach absolvieren muß.

So präsentiert sich Drachenland als Jump'n'Run-Spiel, während das Traumland eher als Actionabenteuer beschrieben werden kann. Neben reichlich Joystick-Action erwartet den Spieler auch ein Rätselmoment, denn um bestimmte Teile des Parks betreten zu können, müssen andere Teile erst von Dämonen gereinigt sein. Theme Park Mystery stellt sich als komplexes Actionabenteuer mit viel Variationsreichtum dar. Das globale Konzept des Spiels ist interessant und abwechslungsreich, und auch die einzelnen Subspiele sind recht orginell gemacht. Alles in allem ein nettes Spiel, das auch im Detail spielbar ist und nicht nur mit augenwischenden Äußerlichkeiten Effekthascherei betreibt.

(hs)

AMIGA DOS Blitzlicht

Name: Theme Park Mystery
Hersteller: Image Works
Vertrieb: Fachhandel
Preis: stand bei Redaktionsschluß noch nicht fest

Positiv:
- mehrere Subspiele
- passable Grafik

Negativ:
- nervige Musik



Atomix

Atomix heißt das neue Spiel von Thalion. Der Sinn des Spiels ist ganz einfach. Es gilt, innerhalb eines Labyrinths verschiedene durcheinander geratene Moleküle neu zu ordnen.

Das hört sich ja eigentlich ganz einfach an, hätten die einzelnen Atome nicht die Eigenschaft, sich nicht beliebig bewegen zu lassen. Ein Atom bewegt sich nämlich immer so lange in eine Richtung fort, bis es entweder auf ein anderes Atom oder ein Mauerstück stößt. Die Steuerung geschieht wahlweise per Maus oder per Joystick, wobei hier die Maussteuerung vorzuziehen ist. Falls man die Zusammensetzung der einzelnen Moleküle nicht kennt (woher auch), muß man nur einen Blick auf die linke Seite des Bildschirms werfen. Dort wird gezeigt, wie das fertige Molekül auszusehen hat. Für jedes Level, sprich für jedes Molekül, hat der Spieler eine bestimmte Zeitspanne zur Verfügung.

Hierin unterscheiden sich auch die verschiedenen Schwierigkeitsgrade. Je höher der Grad, desto weniger Zeit hat man pro Level für ein Molekül. Die ersten Stufen sind ja noch recht einfach zu bewältigen, bei späteren Levels muß man aber erst mal einige Zeit überlegen, bis man überhaupt den Platz herausgefunden hat, an dem das fertige Molekül liegen muß. In ihnen kann der Spieler Punkte sammeln, ohne befürchten zu müssen, daß er kaputtgeht. Atomix ist ein gutes Spiel mit einem friedlichen Spielkonzept und vielleicht einem kleinen Lernwert. Einziger Nachteil ist, daß man die 20 Levels im Easy-Modus sehr schnell durchgespielt hat.

(Andreas Polk/hs)

AMIGA DOS Blitzlicht

Name: Atomix
Hersteller: Thalion
Vertrieb: Fachhandel
Preis: 69,95 DM

Positiv:
- neues Spielprinzip
- gute Steuerung
- einfaches Spielprinzip
- Pause-Modus

Negativ:
- schnell durchgespielt





Terry auf Baron Bonsais Spuren

Cloud Kingdoms

Ja ja, der arme Terry. Da kullert er sich so durchs Leben, und plötzlich wendet sich alles gegen ihn. Der böse Baron von Bonsai hat all seine magischen Kristalle ins Land der Cloud Kingdoms verfrachtet.

Dort will er mit deren Energie Terrys heißgeliebte Wolkenfeen versklaven. Natürlich kann Terry das nicht so ohne weiteres durchgehen lassen. Also macht er sich auf den Weg, der mit vielerlei Gefahren gespickt ist. So kugelt er vorbei an eisbedeckten Flächen, flipperähnlichen Prellwänden, Schubkacheln, saugfreudigen Magneten und Falltüren. Aber nicht nur der Weg zu den Cloud Kingdoms ist beschwerlich, natürlich hat Baron Bonsai Terry auch linke Spießgesellen entgegengeschickt, die ihn aus den Weg schaffen sollen. So begegnet er Billardkugeln, Marienkäfern und allerlei anderem Getier. Das war es dann aber auch schon, was sich da am Bildschirm tummelt.

Natürlich hat sich Terrys Vorhaben schon überall herumgesprochen und seine Freunde haben ihm vielerlei Hilfen auf den Weg gelegt, so zum Beispiel Farbtöpfe, mit denen er Brücken über Abgründe bauen kann, Schilde zum Abwehren von Bonsais Spießgesellen oder Flügel, mit denen er Hindernisse überfliegen kann. Cloud Kingdoms ist ein Spiel, das wenig Neues bietet. Irgendwie ist alles schon einmal dagewesen. Doch trotzdem macht die Hatz auf die Diamanten einen Heidenspaß. Die Levels sind weder zu lang noch zu schwer und bieten viel Abwechslung. Trotz einiger Gemeinheiten, zum Beispiel den Falltüren, denen man nur mit viel Glück entkommt, oder manch toten Ecke, aus der man nicht mehr herauskommt, ein Spielchen, bei dem die Motivation nicht so schnell nachlassen dürfte.

(Timo Siebert/hs)

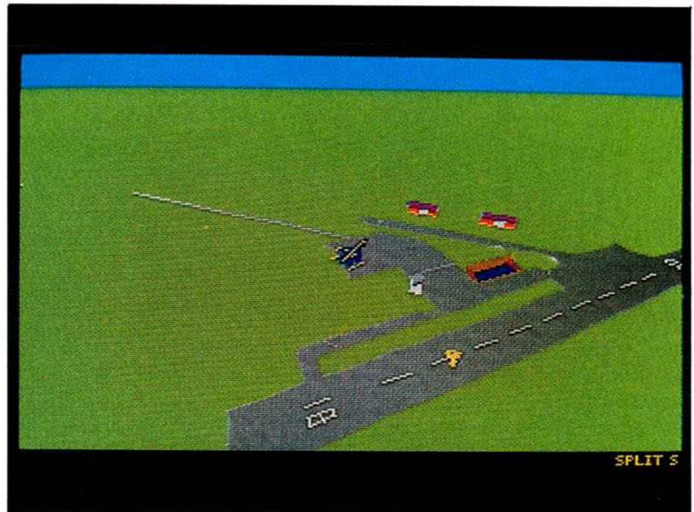
AMIGA DOS Blitzlicht

Name: Cloud Kingdoms
Hersteller: Millenium
Vertrieb: Fachhandel
Preis: 84,95 DM

Positiv:
- spannendes Spielgeschehen
- passable Grafik

Negativ:
- an manchen Stellen viel Glück erforderlich





Wer fliegt so leicht durch Tag und Wind...?

Blue Angels

Fliegen auf dem Amiga – seit Interceptor und dem Flightsimulator hat sich in punkto "Pseudo-Cockpit" einiges getan auf dem Computer. Blue Angels, so heißt eine neue Variation dieses Genres.

Der Spieler wird durch ein Komplett-Training für Flugdemonstrationen gejagt, ehe er, natürlich nur im Spiel, ein Elitemitglied der "U.S.Navy" wird. Geflogen wird mit einer Hornet F/A18. Ob die sich wohl am leichtesten zeichnen läßt? Sie ist ja fast eine Berühmtheit bei Computersimulationen! Wer nicht selbst fliegen will, kann im Zuschauer-Modus den Piloten bei ihren Kunststücken zuschauen.

Blue Angels ist zwar vom Thema her bekannt, hervorzuheben ist jedoch, daß es diesmal nicht gegen irgendwelche Feinde geht, sondern "nur" eine Flugschau zu bestehen ist. In diesem Zusammenhang sei die Anmerkung erlaubt, daß simulierte Flugschauen nicht zu Abstürzen führen – höchstens beim Prozessor. Und tatsächlich gelingt es dieser friedlichen Flugsimulation, den Spieler über längere Zeit zu fesseln. Der hier praktizierte Kunstflug ist mindestens ebenso spannend wie ein Aufklärungsflug in feindliches Gelände.

Blue Angels ist dank des deutschen Handbuchs schnell durchschaubar, die Grafik ist gut; Anflüge werden zum Beispiel zur Beobachtung in einem Kubus dargestellt, der die Flugroutine dreidimensional zeigt. Der Sound ist wie bei andern Simulationen auch digitalisiert.

Blue Angels ist bei Hobby-Fliegern garantiert eine willkommene Bereicherung ihres Software-Bestands.

(jb)



AMIGA DOS Blitzlicht

Name: Blue Angels
Hersteller: Accolade
Vertrieb: Fachhandel
Preis: 84,95 DM

Positiv:
- friedfertige Flugsimulation
- orginelles Programmkonzept

Negativ:
- umständliche Bedienung



PIRATES!

Sind Sie nicht auch fasziniert von den spannenden Seeräubergeschichten, wie man sie von Film und Fernsehen her kennt? Bei dem Spiel Pirates können Sie selbst feindliche Schiffe kapern, Städte plündern und verborgene Schätze suchen.



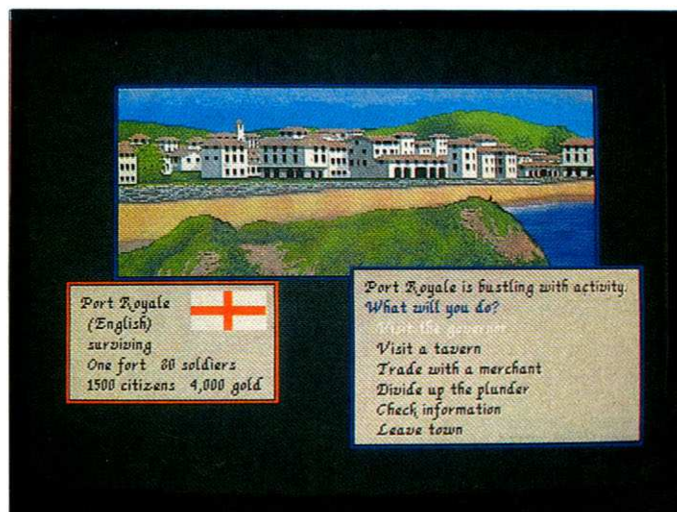
Auf Kaperfahrt in die Karibik...

Wenn man das Wort Karibik hört, denkt man an romantische Kreuzfahrten, an Sonne, Palmen und Sandstrände. Die Simulation Pirates führt uns in diese Breiten, auch wenn es im 17. Jahrhundert dort aufgrund der vielen Schmuggler und Seeräuber etwas stürmischer zuging, als es heutzutage der Fall ist.

Die Seemächte England, Spanien, Frankreich und Holland kämpfen mit allen Mitteln um die Vorherrschaft in der Karibik. An dieser Stelle kommen Sie ins Spiel: Als junger Kapitän eines Piratenschiffs müssen Sie sich beweisen. Da gilt es nicht nur nautische Fertigkeiten zu beherrschen, der junge Seelord muß auch inmitten feindlicher Breitseiten bestehen und seine Fechtkunst im Duell mit dem gegnerischen Kapitän unter Beweis stellen. Anhand einer dem Spiel beigelegten (originalgetreuen) Seekarte können Sie die Fortschritte Ihrer Kaperfahrt kontrollieren. Ferner besteht die Möglichkeit, mit Hilfe eines Astrolabiums (astronomisches Beobachtungs- und Meßgerät) Ihre momentane Position anhand der Sonne zu ermitteln. Es ist weiterhin wichtig, daß Sie auch Ihre Mannschaft im Auge behalten, denn nur zu schnell fängt die Besatzung an zu meutern. Dann lassen sich die Männer nur noch durch den Degen wieder zur Räson bringen. Haben Sie ein Schiff erfolgreich gekapert, können Sie den Segler, nachdem die Ladung übernommen wurde, versenken oder in Ihre Flotte aufnehmen. Es ist somit möglich, eine ganze Armada aufzustellen. Bei weiteren Gefechten kann der Spieler dann bestimmen, mit welchem Schiff er das Gefecht durchführen will. Ferner ist es möglich, seine Flotte in Parteien aufzuteilen, um so verschiedene Aktionen gleichzeitig durchzuführen. Beispielsweise kann man mit einer Gruppe eine Schatzsuche aufnehmen (sofern schon eine Schatzkarte gefunden wurde), während der andere Teil auf Kaperfahrt geht.

Gelangt man in die Nähe einer Stadt, stehen verschiedene Möglichkeiten offen: Zum einen kann man direkt in den Hafen segeln – dies sollte man bei feindlichen Städten vermeiden, sonst eröffnen die durch Forts gesicherten Hafenanlagen das Feuer, was zum Verlust Ihrer Armada führen kann – oder man kann versuchen, die Stadt anzugreifen und auszuplündern. Ferner besteht die Möglichkeit, sich heimlich in die Stadt zu schleichen. In einer Stadt angekommen, kann man den dortigen Gouverneur besuchen, um eventuell einen Geheimauftrag zu bekommen, der Ruhm und Ehre verspricht, oder um Hinweise über den Aufenthalt der vermißten Schwester zu erlangen (die entführte Schwester ist ein Schicksalsschlag, der Sie schon zu Beginn des Spiels trifft). Ein Gouverneur ist zudem in der Lage, Ihre Heldentaten durch Titel und Ländereien zu belohnen – eventuell springt sogar eine Vermählung mit seiner Tochter dabei heraus.

In den Tavernen können Informationen ausgetauscht werden. Hier können unter anderem Schatzkarten, die zu unermeßlichem Reichtum führen, oder Hinweise über die Befestigungen feindlicher Städte erstanden werden. Ferner besteht die Möglichkeit, Handel zu treiben. Kapergüter (und auch Schiffe) können bei dem Händler ver- und eingekauft werden. Eventuelle Schäden an Ihren Schiffen können hier auch ausgebessert werden. Sind Sie der Meinung, genügend Kapital "erwirtschaftet" zu haben, können Sie die Beute unter der Mannschaft aufteilen und sich zur Ruhe setzen.



...werden Spieler entführt, die sich an Pirates versuchen

Pirates ist eine aktionsgeladene Freibeuter-Simulation, die mit abwechslungsreichem Geschehen aufwarten kann. Hat sich Pirates schon seit einiger Zeit auf anderen Computersystemen einer großen Beliebtheit erfreut, so ist nun auch den Amiga-Besitzern dieser Spaß vergönnt. Leider ist bei der Amiga-Umsetzung der Sound etwas dürftig ausgefallen – mehr als ein paar Geräusche sind leider nicht drin. Die Grafiken können sich durchaus sehen lassen, und die Spielmotivation läßt auch nach längerem Spielen nicht nach. Das umfangreiche deutsche Handbuch gibt einen guten Einstieg in das Spielgeschehen. Besonders hervorzuheben ist die Tatsache, daß die Steuerung sowohl per Maus als auch über die Tastatur oder einen Joystick angesprochen wird. Die Firma Microprose, die schon für einige Spitzensimulationen gesorgt hat, hat mit Pirates sicherlich wieder einen Renner in den Amiga-Charts zu verbuchen.

(hs)

AMIGA DOS Blitzlicht

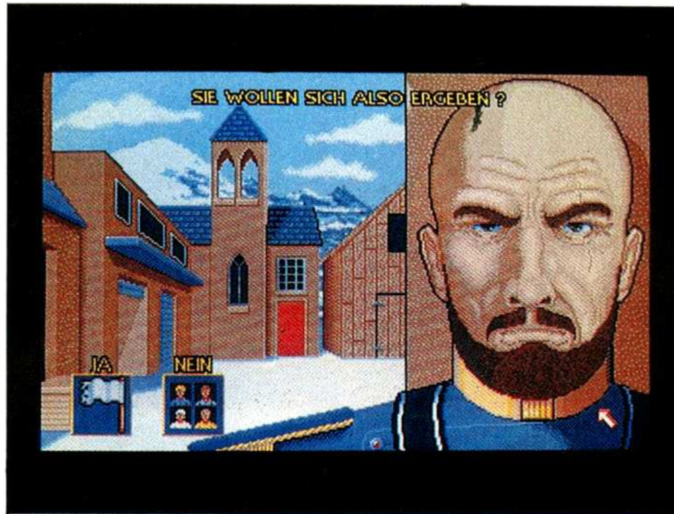
Name: Pirates
Hersteller: Microprose
Vertrieb: Fachhandel
Preis: 89,– DM

Positiv:
- umfangreiches deutsches Handbuch
- abwechslungsreiches Spielgeschehen
- Steuerung wahlweise per Maus, Tastatur oder Joystick
- Installation auf Festplatte möglich (Keydisk-Abfrage)
- Spielstände abspeicherbar

Negativ:
- schwacher Sound

AMIGA DOS URTEIL GUT

GRAFIK	SOUND	MOTIVATION
8	2	9



Keine Angst, bei Midwinter kommen Sie nicht ins Frösteln

MIDWINTER

Mitte des 21. Jahrhunderts geschieht das, was Wissenschaftler heute schon prophezeien. Durch den Menschen ins Ungleichgewicht gebracht, spielt unsere gute Mutter Erde verrückt und wirft mit Naturkatastrophen nur so um sich. Obendrein schlägt noch ein riesiger Meteorit auf der Erde ein, womit die neue Eiszeit perfekt ist.

Die Durchschnittstemperaturen sinken gewaltig. In den kälteren Gebieten setzen Vergletscherungen ein und machen diese Staaten nahezu unbewohnbar. Deren Bevölkerung flüchtet vor den unzumutbaren Lebensbedingungen in äquatornah gelegene Länder. Erloschene Vulkane brechen wieder aus, Erdbeben sind an der Tagesordnung. Und plötzlich hebt sich eine neue Insel aus dem Meer. Wissenschaftler machen sich auf den Weg, die Insel zu erforschen und eventuell neuen Lebensraum zu erschließen. Doch trotz des Chaos, das derzeit auf der Erde herrscht, gibt es immer noch einzelne Besessene, denen die Einsicht fehlt, daß der einzige Weg, diesem Desaster zu entkommen, die Zusammenarbeit aller Menschen ist. Sie versuchen, mit aller Macht die Herrschaft dieser neu entdeckten Insel an sich zu reißen. So auch General Masters, der mit einem hohen Truppenaufgebot versucht, die Mission der "Free Villages Peace Force", deren Führung Sie übernehmen, zum Scheitern zu bringen.

Ihre Aufgabe ist es nun, das Hauptquartier von Masters zu erreichen und einzunehmen. Da Ihnen Masters Truppen zahlenmäßig überlegen sind, kommt ein Kampf "Auge in Auge" nicht in Frage. Ihr Team besteht aus 32 Mitgliedern, die quer über die ganze Insel verstreut leben. Der erste Teil des Spiels besteht darin, diese Leute zusammenzutrommeln, um ein wirkungsvolles Handeln gegen Masters zu planen. Alle 32 Mitglieder haben unterschiedliche Eigenschaften. So eignet sich der eine zum Beispiel hervorragend für Sabotageaktionen, während der andere im Schießen sehr geübt ist und ein dritter das Schneebuggyfahren perfekt beherrscht. Diese Charaktere sind jetzt strategisch einzusetzen, um die Basis des Generals, welche im Südosten der Insel gelegen ist, einnehmen zu können. Wer jetzt denkt: "Hoho, 21. Jahrhundert, mal sehen, mit was für 'nem Hightech-Zeugs wir da durch die Gegend wandern", wird schwer enttäuscht werden. Die winterlichen Fortbewegungsmittel sind die alten geblieben. So schiebt man seinen goretexgepolsterten Body mit Skiern oder per Schneebuggy durchs Land. Und wer Glück hat, findet sogar einen Drachenflieger, und wer noch etwas mehr Glück hat, kriegt das Teil sogar noch in die Luft, und wer am meisten Glück hat, bleibt dann auch oben.

Natürlich bleibt man auf seinem Weg zu Masters Basis nicht ungestört. So trifft man immer wieder auf feindliche Objekte, seien es nun Flugzeuge, die mit Bomben nur so um sich schmeißen, oder feindliche Laster, welche nichts anderes zu tun haben, als einen in Grund und Boden zu fahren. Bei einer Konfrontation gibt es immer zwei Möglichkeiten der Verteidigung. Entweder man vernichtet den Gegner durch Beschuß (mit einem durchschlagkräftigen Gewehr, Granaten oder Missiles), oder man benutzt den Bleifuß und flüchtet in entgegengesetzter Richtung. Zieht man nun gemütlich durch die fraktal gestaltete Landschaft, so möchte man sicher wissen, wo man sich denn eigentlich befindet. Dazu dient einerseits die Darstellung im Display, oder man schaut sich eine Karte der ganzen Insel an, auf der auch verschiedene Siedlungen und anderes angegeben sind. Mit der Option "Leute" kann man seine Teammitglieder ausfindig machen. Ein sehr schöner Einfall ist auch die Option "Relief", mit der der aktuelle Abschnitt der Karte plastisch dargestellt wird, so daß man Höhenzüge und Täler erkennt. Dies kann sehr nützlich sein, um im Schutz der Täler den Feind zu umfahren. Hat man einen Ort erreicht, so kann man Munition nachladen oder etwas essen, um Kraft zu gewinnen. Um dem Feind so wenig Gutes wie möglich zu tun, kann man den besuchten Ort dann auch in die Luft sprengen, so daß er nur noch Ruinen vorfindet und in die Röhre guckt.

Midwinter ist in erster Linie ein Strategiespiel mit einigen Actioneinlagen. Im wesentlichen besteht das Spiel darin, von einem Ort zum anderen zu gelangen, dort ein bißchen "herumzuwuseln" und seine Leute mitzuschleppen. Dies kann auf Dauer nicht begeistern. Zwar ist es ein Genuß fürs Auge, durch die schneebedeckte Landschaft zu ziehen, aber sobald es richtig zur Sache geht, wird das Spiel zu verwirrend, zu komplex und der Bildschirm zu unübersichtlich. Fazit: Nur für eingefleischte Fans dieses Genres der (Action-)Strategiespielchen zu empfehlen.

(Timo Siebert/hs)



Eiskalte Gegner heizen Ihnen kräftig ein

AMIGA DOS Blitzlicht

Name: Midwinter
Hersteller: Microprose
Vertrieb: Fachhandel
Preis: 89,– DM

Positiv:
- gute Spielidee
- gelungene Grafik
- umfangreiche, deutsche Bedienungsanleitung

Negativ:
- verwirrender Spielablauf
- unübersichtlicher Aufbau



Das Spiel kombiniert verschiedene Spieltypen. Einmal ist da die Steuerung des Raumschiffs, das man besitzt. Es handelt sich dabei um Actionsequenzen (bei Angriffen von feindlichen Raumschiffen) und Passagen, in denen man etwas Fingerspitzengefühl benötigt, zum Beispiel beim Andocken an Raumstationen. Allerdings schafft das sogar der absolute Anfänger, so daß einem Frust und Haareraufen erspart bleiben. Zu dem Raumschiff ist man übrigens auf sehr interessante Art und Weise gekommen, hier setzt nämlich die Story an: Auf einem Routineflug des Kreuzers, mit dem Sie unterwegs sind, entdecken Sie ein verlassenes Raumschiff.

Sie bekommen die Aufgabe, es zu untersuchen. Als Sie sich gerade in dem Schiff umsehen und feststellen, daß alles voll funktionsfähig ist, wird Ihr Mutterschiff von den Manchis angegriffen und zerstört. Das Schiff, in dem Sie sich gerade befinden, wird nicht beachtet. Innerhalb von fünf Minuten werden Sie also zum stolzen Besitzer eines Raumschiffs.

Zur Steuerung des Schiffs ist zu sagen, daß zwei verschiedene Modi zur Auswahl stehen. Bei einem (Cruise Flight) verhält sich das Raumschiff relativ ähnlich wie ein Flugzeug. Im anderen Modus (Newtonian Flight) wird die Schwerelosigkeit berücksichtigt, die Steuerung ist relativ schwierig. Die Raumschiffe werden in Vektorgrafik mit ausgefüllten Flächen dargestellt und bewegen sich angenehm schnell über den Bildschirm. Bei Gefechten hat man deswegen auch recht viel Spaß. Hat man erfolgreich an eine Raumstation angedockt, ändert sich das Spielprinzip. Man sieht den Raum, in dem man sich befindet, in Aufsicht und kann darin umhergehen, Personen ansprechen und Dinge unter die Lupe nehmen. Die Grafik ist hier etwas ruckelig, was jedoch nicht weiter stört. Hier kann man handeln, Dinge in Erfahrung bringen, sein Schiff reparieren, Waffen kaufen, an der Bar etwas trinken und sogar an herumstehenden Automaten videospielen. Hat man dort alles erledigt, verläßt man die Station wieder. Sie haben jetzt verschiedene Möglichkeiten. Sie können entweder innerhalb der Galaxie bleiben, in der Sie sich befinden, oder Sie können die Galaxie verlassen und in eine andere fliegen.



Eine Karriere als Weltraumhändler erwartet Sie

Space Rogue

Space Rogue ist ein Spiel, das in die Fußstapfen von Elite tritt. Ziel ist es, zu handeln, Geld zu verdienen, aufzusteigen und die Manchis, eine Spezies fieser Aliens zu vernichten.

Um sich fortzubewegen, schaltet man in den Navigationsmodus um. Dort hat man eine Karte der gesamten Galaxie, in der man sein nächstes Ziel festlegen kann. Hat man sein Ziel festgelegt, wird der Kurs im Bordcomputer gespeichert, man braucht nichts weiter zu tun, als den Autopiloten einzuschalten. Das Raumschiff bewegt sich dann auf der Karte (und im Zeitraffer) zum angegebenen Ziel. Es kann sein, daß Sie unterwegs angegriffen werden, es bleibt Ihnen dann nichts anderes übrig, als auf Cockpitsicht umzuschalten und den Kampf zu führen.

Sind Sie schließlich an Ihrem Ziel angekommen, müssen Sie nur noch von Hand an die Raumstation ankoppeln. Die Bewegung zwischen verschiedenen Galaxien erfolgt durch eine Art "Zeittunnel", man muß dort einiges Geschick beweisen, um wirklich die gewünschte Galaxie zu erreichen. Die Grafik von Space Rogue ist zwar nicht toll, aber sie erfüllt ihren Zweck sehr gut, man kann sich nicht beschweren. Allerdings sieht es bezüglich des Sounds ganz mager aus. Es gibt nämlich keine Titelmelodie, und auch im Spiel ertönen höchstens bei Gefechten oder zu stürmischen Andockversuchen mal ein paar Geräusche (als mehr kann man das nicht bezeichnen). Das Fehlen von Sound und die "normale" Grafik mindern allerdings nicht den Spielspaß, da Space Rogue eher ein Handelsspiel ist, ist Grafik nicht so sehr gefragt. Bei Kämpfen kommt es auch mehr auf die Geschwindigkeit denn auf 4096 Farben im Overscan-Modus an. Auch das mitgelieferte Material ist ausreichend. Neben einigen Anleitungsheftchen befindet sich auch ein Bastelbogen zum Nachbasteln des Raumschiffs im Karton.

Wer eine echte Alternative zu Elite sucht, hier wird er fündig. Die Raumflugsequenzen des Spiels können sich in puncto Programmqualität und Schnelligkeit mit Elite messen. Die Rollenspielelemente, die den Spieler an Bord der verschiedenen Raumstationen erwarten, sind sozusagen das Salz in der Suppe und sorgen für Abwechslung. Space Rogue ist wie geschaffen für Hobbyastronauten, die abends gerne einmal die fernen Spiralarme der Galaxis unsicher machen.

(Andreas Polk/hs)



Ein interessanter und gefährlicher Job

AMIGA DOS Blitzlicht

Name: Space Rogue
Hersteller: Origin
Vertrieb: Fachhandel
Preis: 99,– DM

Positiv:
- gute Mischung aus Adventure, Action und Handelssimulation
- gutes Handbuch

Negativ:
- magerer Sound
- schlechtes Scrolling

AMIGA DOS URTEIL GUT





Attacke auf die Insektenaliens - Warhead

Warhead

Warhead ist ein Flugsimulator im Weltraum. Ziel ist es, Aliens zu vernichten, die Millionen von Menschen auf der Erde getötet haben.

Dem Spieler steht ein Raumschiff zur Verfügung, das mit verschiedenen Waffensystemen ausgestattet ist. Man befindet sich in einem Cockpit mit Headup-Display, wie in einem Kriegsflugzeug. Den Unterschied zu einem Flugzeug erkennt man spätestens nach dem Starten von der Basis. Die Steuerung ist nämlich ausgesprochen ungewohnt, die Schwerelosigkeit recht gut imitiert. Außerdem wird mit der Maus gesteuert, was die Sache nicht gerade erleichtert. Diese Art der Steuerung, so realistisch sie auch ist, bringt den Spieler zur Verzweiflung, so daß es nicht außergewöhnlich sein dürfte, wenn das Spiel nach dem zehnten mißglückten Andockversuch an die Basis im Mülleimer landet.

Das Handbuch trägt auch nicht gerade zur Beruhigung des Spielers bei, denn es ist sehr mager. Auf der Suche nach Tastenbelegungen kommt man um vieles Blättern nicht herum, und ein Hinweis auf die Taktik zur Durchführung der Aufgabe wird auch nicht gegeben. Man steht als Spieler also ganz alleine da. Zu der wenig ausgefallenen Spielhandlung und der schwierigen Steuerung kommt also noch eine mäßige Dokumentation, was das Spiel nicht gerade brauchbarer macht.

Die auf der Verpackung des Spiels garantierten 30 Stunden Spielvergnügen können leicht in 30 Stunden Dauerfrust ausarten. Sound gibt es bei dem Programm übrigens außer Explosionsgeräuschen und einer (guten) Titelmelodie nicht. Die Grafik hingegen ist gelungen. Es wurde auch auf Details geachtet, so zum Beispiel das Flimmern des Message-Bildschirms (wie im Fernsehen). Das Programm hat gute Ansätze, in der vorliegenden Form ist es aber wenig motivierend.

(Andreas Polk/hs)

AMIGA DOS Blitzlicht

Name: Warhead
Hersteller: MPH
Vertrieb: Fachhandel
Preis: 84,95 DM

Positiv:
- detailreiche Grafik

Negativ:
- schlechtes Handbuch
- schwierige Steuerung



Softwarenoten und wie sie zustandekommen

Daß Ihre AMIGA DOS bei den Spiele-Tests ein Urteil vergibt, haben Sie selbst sicherlich schon gesehen. Wie dieses Urteil funktioniert, das soll hier noch einmal im Detail erklärt werden.

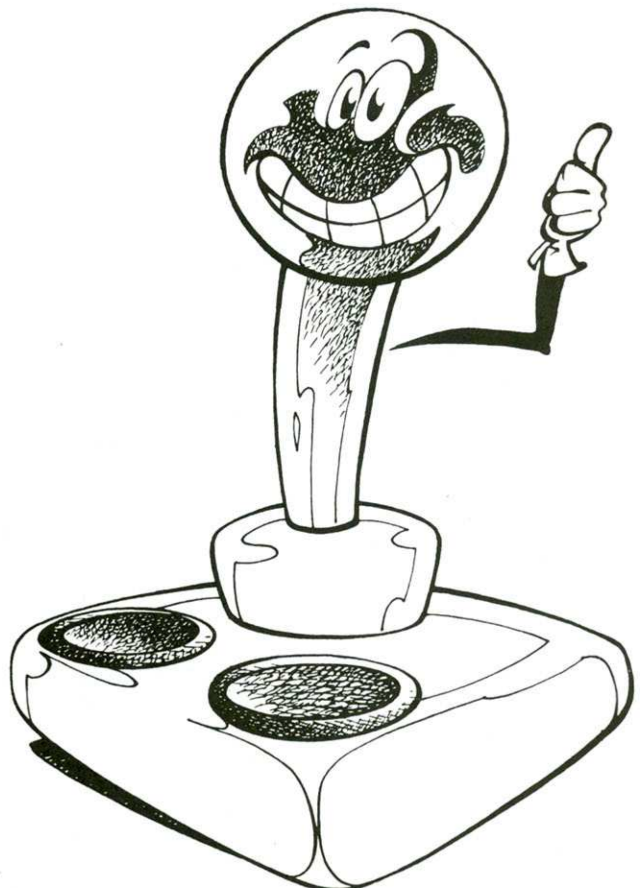
AMIGA DOS beurteilt jedes Spiel anhand verschiedener Kriterien: Die Grafik, die Musik und den Spielspaß. Die Noten dafür werden in der kleinen Urteilsbox, die am Ende jeder Review zu finden ist, unter den Oberbegriffen Grafik, Sound und Motivation dargestellt. Die Note selbst richtet sich nach der Höhe des wiedergegebenen Wertes. Je höher, desto besser. Zu diesen Noten gesellt sich ferner ein individuelles Urteil des Testers. Eine vierte Note also, die allerdings nicht wiedergegeben wird.

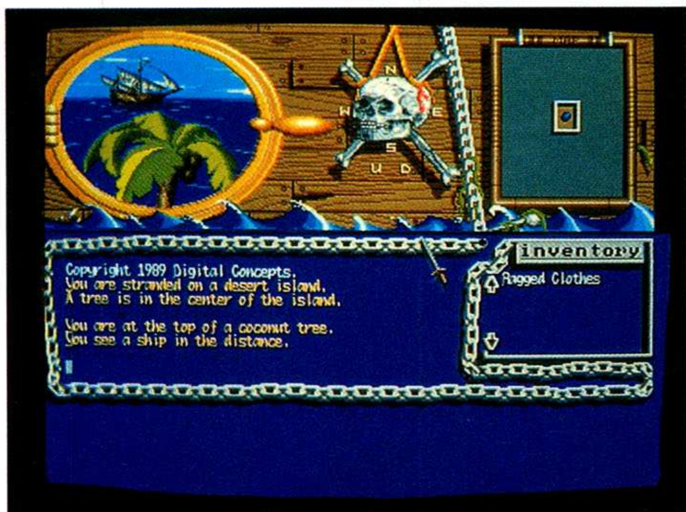
Aus diesen vier Werten bildet sich schließlich das Gesamturteil:

- **Sehr gut**
- **gut**
- **o.k.**
- **geht so**
- **schlecht**



Die Bedeutung dieser Bewertungen erklärt sich von selbst. Uns bleibt noch, Ihnen viel Spaß mit dem AMIGA DOS Spieleteil zu wünschen.





Schiffbruch, Schätze und Piraten – ein Textadventure für Hobbyfreibeuter

Island of lost Hope

Seefahrt, Schätze, Piraten, Haie und einsame Inseln; nein, es erwartet Sie nicht etwa eine Neuinszenierung des scharlachroten Piraten, sondern eine Renaissance des Textadventures.

Das Spielkonzept ist einfach. Dem Spieler werden Beschreibungen der Schauplätze ausgegeben, aus denen er alles Wesentliche über die aktuelle Situation erfährt. Er muß nun auf die beschriebene Szene reagieren, indem er seinerseits Kommandos für das weitere Vorgehen eintippt. Island of lost Hope basiert zwar auch auf diesem Konzept, nutzt allerdings die Möglichkeiten des Amiga besser aus als ein reines Textadventure. So gibt es neben den erläuternden Texten auch Geräuscheffekte und Bilder, die die Örtlichkeiten illustrieren.

Die Story ist nett aufgemacht. Der Spieler strandet auf einer kleinen, menschenleeren Insel. Als dann ein Piratenschiff auftaucht, gilt es zum einen, von der Insel zu entkommen, und zum anderen, den blutrünstigen Piraten nicht schutzlos in die Arme zu laufen. An Bord des Freibeuterschiffes ist es unbedingt notwendig, unerkannt zu bleiben und die Geheimnisse, die sich in der Kapitänskajüte und dem Laderaum verbergen, zu lüften. Während des Spiels zeigt eine kleine Karte auf dem Monitor dem Spieler immer genau an, wo er sich gerade befindet. Nackte Textadventures sind überholt. Dies beweist nichts deutlicher als der Niedergang der Infocom-Adventure-Spezialisten. Dieses Spiel hat zweifelsohne seine Reize für alle Abenteuerfreunde. Wer das Spielgenre nicht mag, wird wohl auch mit diesem Programm nicht viel anfangen können.

(hs)

AMIGA DOS Blitzlicht

Name: Island of lost Hope
Hersteller: Digital Concepts
Vertreiber: Fachhandel
Preis: 84,95 DM

Positiv:
- unterhaltsame Story
- moderater Schwierigkeitsgrad

Negativ:
- Grafiken zu klein





Wird es Ihnen gelingen, Großkalif von Khalaan zu werden?

Khalaan

Vier Kalifen herrschen zur Zeit im Lande Khalaan. Wie üblich, will jeder Großkalif werden. Einer Prophezeiung zufolge kann jedoch nur derjenige Großkalif werden, der den fremden Eindringling abwehrt.

Khalaan ist ein Kalifat zwischen Tigris und der Wüste von Lut. Natürlich ist jeder Kalif bestrebt, die Macht an sich zu reißen. Und zu diesem Zweck ist ja lediglich der Eindringling mit seinen Truppen zu besiegen. Dazu ist natürlich eine geldverschlingende Armee notwendig, die kurzerhand rekrutiert wird. Aber Armeen sind nur ein Mittel, um sich durchzusetzen. Mindestens ebenso wichtig ist es jedoch, Informationen über seine Gegenspieler einzuholen. Dies geschieht mit Hilfe von Spionen, die sich allerdings ihre Dienste teuer bezahlen lassen. Auf der anderen Seite kann man durch geschicktes Meucheln seiner "Mitbewerber" so manches Goldstück einsparen, das die Armeen bei einer Offensive verschlungen hätten. Der Spieler muß nun versuchen, durch den Bau und die Eroberung von Städten und Festungen einerseits seine Reserven zu erhöhen und andererseits den Gegner entscheidend zu schwächen. Bei aller Konzentration darf jedoch die Situation im eigenen Land nicht vernachlässigt werden. Bei aufkommenden Unruhen sinken die Steuereinnahmen, was sich negativ auf die Bezahlung der Armeen auswirken kann. Spenden an das Volk helfen hier, den sozialen Frieden zu erhalten.
Hier kommt nun die Wunderlampe ins Spiel, die jeder ambitionierte Kalif an sich reißen sollte, wenn sich eine passende Gelegenheit ergibt. Sie stellt sozusagen die Geheimwaffe des frühen Mittelalters dar.

(mm)

AMIGA DOS Blitzlicht

Name: Khalaan
Hersteller: Rainbow Arts
Quelle: Rainbow Arts
Preis: 89,– DM

Positiv:
- interessante Thematik
- ausführliches Handbuch

Negativ:
- mäßige Grafik
- kaum diplomatische Optionen
- nervige Musik



AMIGA DOS - Spieletips

Oldies but Goldies. Elaborate, auf die diese Umschreibung zutrifft, findet man nicht nur im weiten Feld der Schlager. Mehr als ein Amiga-Spiel hat sich inzwischen den Titel "Klassiker" verdient. Und natürlich werden wir Sie nicht nur mit heißen Spieletips zu neuen Programmen versorgen, sondern auch die "Goldies" und "Evergreens" berücksichtigen.

Es gibt einen Cheatmode für alle, die bei "Shadow of the Beast" bereits nach kurzer Zeit im Boden versinken. Sobald das Titelbild erscheint, drückt man einfach den Joystickknopf und die linke Maustaste. Man muß beide so lange gedrückt halten, bis die Meldung kommt, daß die Disks gewechselt werden sollen. Wenn man jetzt den Joystickknopf und das Ohr der Maus wieder losläßt, steht dem Erforschen der verschiedenen Levels nichts mehr im Wege.

Ein paar Kurztips als Hors d'oeuvre

Ebenfalls zu Beast kommt noch ein eigener Tip von uns, den wir aber ohne den Cheat herausgefunden haben: Nach dem Start sollte man zunächst nach rechts und in den Baum. Der erste Weg sollte dann nach unten führen, wo man sich einen Schlüssel und eine Extrawaffe beschaffen kann, mit der man dem Zwischenbösewicht den Garaus machen kann. Hat man ihn erledigt, geht es an der nächsten Treppe nach oben, wo man einen Hebel umlegt und sich einen weiteren Schlüssel besorgt. Weiter unten gibt es dann einen Power Punch für das Schlußmonster in der Höhle.

Man verläßt das Labyrinth durch den Brunnen und wendet sich nach rechts. Sollte es im Schloß duster sein, liegt das daran, daß man die Fackel, die außen an der Schloßmauer hängt, vergessen hat.

Hilfe für Leitungsbauer

Die Passwörter zu Pipemania lauten im einzelnen:

GLICK
TICK
DOCK
OOZE
BALL
WILD

Spielt man Pipemania zu zweit, hat sich folgende Taktik bewährt: Ein Spieler geht an den Start und versucht, den Start so schnell wie möglich mit einem der Auffangbehälter zu verbinden und auf diese Weise Zeit zu schinden. Der andere baut vom Ende ausgehend ein möglichst großes Stück Leitung, an das der Start nur noch angeschlossen zu werden braucht. Auf diese Weise kann man fast jedes Level schnell hinter sich bringen.

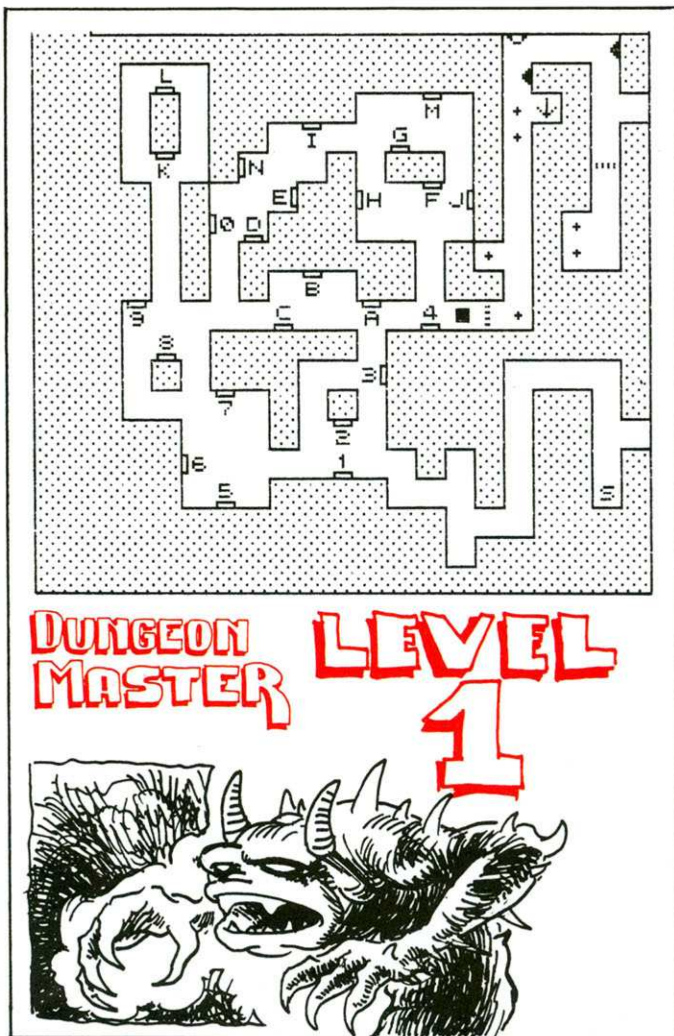
Beach Volley

Gibt man hier während des Spiels »DADDYBRACEY« ein und drückt anschließend die Taste F1, kommt man einen Level weiter.

Dungeon Master

Zu den Karten, die Sie auf diesen Seiten sehen, finden Sie hier die Zeichenerklärung.

Im ersten Level gibt es noch keine Finsterlinge. Hier erwarten Sie nur die Helden, die Sie in Ihre Party aufnehmen können. Sie finden dort folgende Charaktere:



Dungeon Master: Karte Level 1

1 - Iaido   2 - Zed
3 - Elija   4 - Halk
5 - Chani   6 - Hawk
7 - Boris   8 - Alex
9 - Nabi   0 - Sonja
A - Syra   B - Gando
C - Linfas   D - Leyla
E - Wuuf   F - Wutse
G - Leif   H - Tiggy
I - Stamm   J - Daroou
K - Hisssssa   L - Gothmog
M - Azizi   N - Mophus
\+ - Bezeichnet wertvolle Gegenstände
K - Schlüssel

Die restlichen Karten folgen in den nächsten Ausgaben der AMIGA DOS.

In dieser Ausgabe wollen wir unsere kleine Reihe mit Lösungen zu Sierras Space-Quest-Reihe mit dem dritten und vorläufig letzten Teil beenden. Sollte sich Sierra entschließen, noch einen weiteren Teil zu entwickeln, werden wir für Sie natürlich am Ball sein.

Roger Wilco und die Jungs von Andromeda.

Nach der erfolgreichen Flucht aus Space Quest II befindet sich unser Freund Roger auf einem riesigen Mülltransporter. Da das Gefährt nicht sonderlich komfortabel ist, heißt es zunächst einmal, das Raumschiff zu verlassen. Zu diesem Zweck erscheint es sinnvoll, sich auf die Suche nach einem Gefährt zu machen, mit dem man sich auch im Weltraum unbescholten fortbewegen kann. Das kleine Schiff im nordöstlichen Teil des Frachters sieht sehr vielversprechend aus, ist aber leider im Moment weder betretbar, noch einsatzbereit. Roger findet es, in dem er durch das Auge des Battlebots klettert (enter eye). Um es für unsere Zwecke nutzbar zu machen, benötigen wir einige Teile.

Das erste finden wir gleich an unserem Startpunkt, es ist der Warp Motivator. Da er nicht ohne weiteres bewegt werden kann, müssen wir uns etwas einfallen lassen. Der Kran an der Decke könnte sich dabei als nützlich erweisen. Um ihn zu erreichen, benutzen wir zunächst die Förderanlage im südöstlichen Teil des Frachters. Im richtigen Moment muß man allerdings absprin-



Dungeon Master: Karte Level 2

gen, da Roger sonst ein wenig zerkleinert wird (stand, jump track). Wir balancieren nun westwärts und erreichen auf diese Weise den Kran, dessen Bedienung, ist er erst einmal geentert, uns vor keinerlei Probleme stellt (press claw). Nachdem der Motivator jetzt an der richtigen Stelle ist, können wir die Suche fortsetzen. Die Maschine kann nur im Raum mit dem Droiden verlassen werden.

Dieser etwas ungewöhnliche Weg führt Roger aber direkt zum nächsten Teil. An der Westwand des durch den Schacht zugänglichen Raumes befindet sich nämlich ein Reaktor. Verlassen wird die Höhle mit Hilfe der Leiter, die wir natürlich mitnehmen. Die ansässigen Ratten möchten leider "ihren" Reaktor behalten, so daß Sie ihn sich dort, wo Sie ihn schon das erste Mal gefunden haben, wiederholen müssen.

Es hilft nichts, wir müssen uns den Reaktor noch einmal besorgen. Die Leiter wird wieder mitgenommen und diesmal nach den diebischen Ratten Ausschau gehalten (look for rats), die dieses Mal nicht in Erscheinung treten. Das Kabel von der Wand wandert selbstverständlich auch in unsere Taschen. So ausgerüstet, steht der Inbetriebnahme des Schiffs nun nichts mehr im Wege: Mit der Leiter rückt die Luke in erreichbare Nähe (use ladder, climb), und der Weg ins Schiff steht offen (enter hatch). Im Schiff installieren wir den Reaktor, wobei wir das zu kurze Kabel mit unserem überbrücken. Der Computer teilt uns nun mit, ob das Schiff betriebsbereit ist. Falls ja, setzen wir uns in den Pilotensitz, der auch noch etwas Geld beherbergt.

Die Fortsetzung der Space-Quest-Lösung folgt in der nächsten AMIGA DOS.

Nach den Tips zu Journey, die Sie in der AMIGA DOS 2/90 fanden, wurde es recht ruhig um Praxis, den Zauberer, und seine Reisegefährten. Nun gibt es neue Tips zu diesem Spiel, in denen Praxis seine Reiseerzählung wieder aufnimmt. Diese Tips knüpfen nahtlos an den bisher beschriebenen Weg an.

Neue Tips zu Infocoms Journey

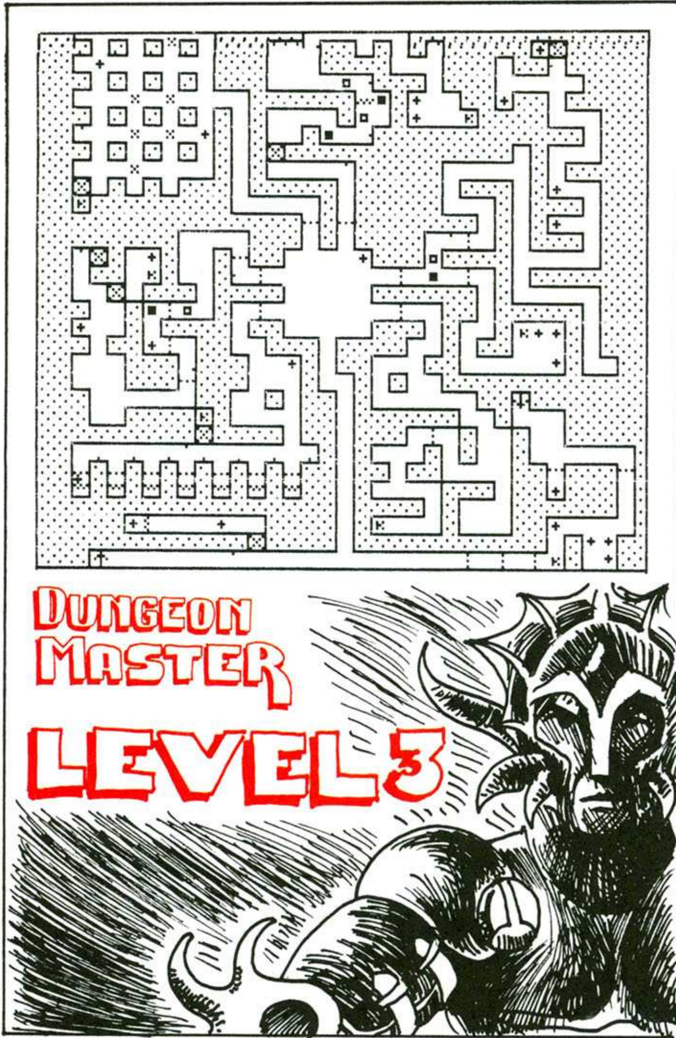
Als wir uns wieder auf den Weg machen wollten, um die zwei Steine zu finden, überreichte Astrix Praxis noch eine Substanz, die er mit den Essenzen mischen könnte. Aber nicht jede Mischung funktionierte, und so war Vorsicht geboten. An der Weggabelung entschlossen wir uns, auf dem schnellsten Weg zum Bern i Fen zu wandern. Schon kurz nachdem wir die Höhle betreten hatten, sahen wir einen Schatten, der sich hinter uns bewegte. Wir stellten uns ihm, und zu unserer größten Überraschung erkannten wir, daß Hurth sich in diesem Schatten verbarg. Er erzählte uns, daß sein Vater, Agrith, sein Leben unter Verlust seines eigenen gerettet hatte. Sein Gedicht über Cedrith gab uns auch die ersten Hinweise auf einen der gesuchten Steine.

Wir begaben uns eine Ebene tiefer, auf der Hurth aber einige Orcs bemerkte. Seine intensive Suche förderte aber einen Weg zutage, der um sie herum führte. Wir folgten diesem Weg und hielten uns dann nach rechts. Nach kurzer Zeit standen wir vor einer verschlossenen Tür, die mit Runen beschriftet war. Praxis übersetzte die Runen nur ungefähr, da er sich über die Bedeutung von einigen nicht richtig klar war. Praxis ungefähre Übersetzung lautete: "Please only say gate open to enter storageroom." Nach einiger Zeit kam ihm dann aber die zündende Idee. Er stellte fest, daß er die Runen von der falschen Seite her übersetzt hatte. Er las sie von rechts nach links und kam zu dem Ergebnis, daß er nur "please" zu sagen brauchte. Das Wort "lorem" brachte dann auch den gewünschten Erfolg.

Der Raum barg reiche Schätze, die vor allem für Praxis interessant waren. Nachdem wir alles an uns genommen hatten, kehrten wir zur Kreuzung zurück. Es half nichts, wir mußten an den Orcs vorbei. Um sie zu überraschen, ließ Praxis zunächst die Erde beben. Anschließend hob er sich selbst in die Luft, um den anstürmenden Orcs zu entgehen. Wir entschieden uns für den breiten Pfad, der aber schon nach kurzer Zeit immer schmaler wurde. Die Lage war beängstigend. Vor uns rauschte irgendwo machtvoll Wasser, hinter uns lärmten die Orcs. Bald landeten wir in einer großen Grotte, die sich zu unserem Pech als Sackgasse herausstellte. Das Rauschen des Wassers wurde ohrenbetäubend. Es blieb keine andere Alternative, als uns dem Kampf zu stellen. Während Praxis sich darauf konzentrierte, mittels Zauber die Erde beben zu lassen, fingen wir an, mit den Orcs zu kämpfen. In dem harten Gefecht, das sich über mehrere Runden erstreckte, wurde Bergon verletzt. Praxis war gerade noch rechtzeitig mit seinem Zauber soweit. Um uns brach ein Chaos los. Wassermassen stürzten auf uns ein und trugen uns durch einen langen Tunnel, über einen Wasserfall zum Ufer eines Sees. Es war ein wahres Wunder, daß wir keine größeren Schäden erlitten und daß vor allem die Ausrüstung heil und beisammen blieb. Aber Bergon war sehr stark verwundet, und der Sturz über den Wasserfall hatte ein übriges zum schlechten Gesundheitszustand unseres Anführers beigetragen. Um die Wunde zu heilen, mischte Praxis seine Reagenz mit Wasseressenz und probierte diese Mischung erfolgreich an Bergon aus.

An dieser Stelle wurde mir klar, daß es für mich vielleicht noch einmal von großem Nutzen sein könnte, wenn ich wüßte, wie Praxis Essenzen aussehen, falls er einmal ausfallen sollte. Ich hatte nämlich festgestellt, daß jeder Zauberspruch an Praxis' Händen ein wenig Pulver zurückließ, dessen Körnung und Farbe ich mir merkte.

Wir gingen weiter, betraten den Tunnel und folgten dann Geräuschen, die aus dem linken Gang zu hören waren. Wir kamen in ein altes Bergwerk, dessen Wände eine Menge von Astrix' Reagenz bargen. Bergon riet jedoch davon ab, die Wände zu berühren, und ich glaube, daß er damit Recht hatte. Wir folgten den Geräuschen, die aus einem neueren Bergwerk zu uns drangen. Wir gingen durch eine Felsspalte, sahen uns um, konnten aber nichts Besonders entdecken. Als wir zurückgingen, hörten wir das Geräusch aber sofort wieder, und so durchschritten wir die Spalte ein weiteres Mal.



Dungeon Master: Karte Level 3

Nachdem wir die Wände erneut betrachtet hatten, erschien diesmal ein Bergarbeiter mit einem Sack vor uns.

Der Arbeiter erzählte uns einige interessante Dinge über die Orcs und das Bergwerk. Bei einer Vorführung seiner Arbeit fielen ihm einige Stückchen rotes Gestein aus seinen Händen, die ich unbemerkt an mich nahm. Von der Treppe folgten wir zunächst dem Crude Path. In einer Kammer befand sich ein Riß im Boden, der weit in die Tiefe führte. Wir ließen Hurth in die Tiefe hinab, der trotz Seil den Boden nicht erreichte. Er war aber so mutig und sprang. Unten fand er neben einem Totenkopf einen Schlüssel, den er an sich nahm, und kletterte wieder nach oben.

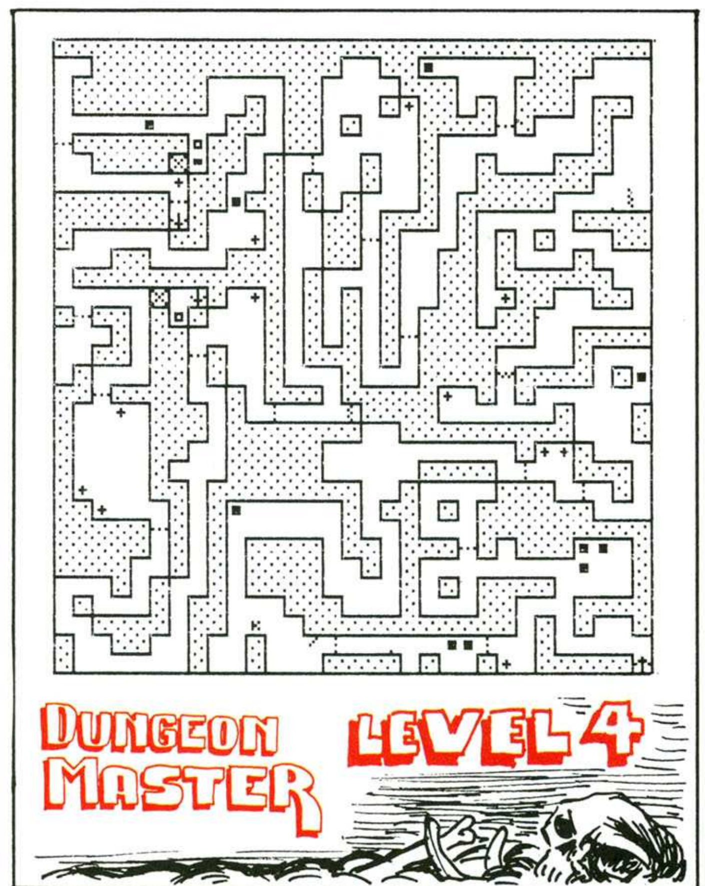
Wir entschlossen uns, zur Weggabelung zurückzukehren. Der prunkvolle Pfad führte uns zu einer verschlossenen Tür, die sich aber mit Hurths eisernem Schlüssel öffnen ließ. Hinter der Tür befand sich ein großes Gewölbe, in dem ein großer eiserner Kasten stand. Die Runen konnte keiner von uns entziffern, und so entschloß sich Praxis, seine Reagenz mit Feueressenz zu mischen. Er wendete dieses Gemisch auf das Gewölbe an, und der Kasten verschwand. Er hinterließ einen Sarg mit Cedriths Leichnam. Einen der gesuchten Steine, den Spirit Stone, trug Cedrith um den Hals.

Wir gingen zweimal zurück und gelangten so in einen undurchdringlichen Wald. Als wir einen im Gestrüpp verendeten Fuchs fanden, entschloß sich Praxis, eine Feuerkugel zu zaubern, die uns den Weg frei machte. Am nächsten Morgen war Praxis verschwunden. Aus seinen Berichten habe ich auch seine Erlebnisse niederschreiben können. Praxis wurde von einem sprechenden Baum geweckt, den er über seine Umgebung ausfragte. Danach entschied er sich für den Weg zur Burgruine und überquerte eine alte Brücke, die hinter ihm zusammenbrach. Da Praxis nun in der Burg eingeschlossen war, erforschte er sein Gefängnis näher. Oben auf dem Turm traf er den sprechenden Baum wieder, der gar kein Baum war, sondern ein Mensch namens Umber. Umbers Bitte, ihn begleiten zu dürfen, nahm er mit Freuden an. Da er sich aber um uns sorgte, entschloß er sich, uns ein Zeichen zu geben. Er zauberte einen großen Regen, der als Zeichen von Bergon erkannt wurde.

Wir machten uns also auf, Praxis zu suchen. Als Hurth Atembeschwerden bekam, konnte Esher ihn mit dem in der Höhle gefundenen Hawkbane heilen. Auf dem weiteren Weg trat Bergon in eine von den Orcs aufgestellte Falle. Jede Hilfe kam für ihn zu spät, da die Orcs schon heraneilten und ihn mitnahmen. Wir folgten den Orcs in ihr Camp. Gegen Hunderte von Orcs hatten wir aber keine Chance, so daß wir auf einen anderen Weg sinnen mußten, Bergon zu helfen.

Ich erinnerte mich daran, daß wir den Bergarbeiter beim ersten Mal nicht gesehen hatten, und setzte das rote Gestein ein. Tatsächlich wurde ich unsichtbar und konnte so Bergon befreien. Da das Gestein bestimmt noch von Nutzen für uns sein konnte, rannten wir offen durch das Lager zurück und sparten so ein wertvolles Zaubermittel.

Praxis stieg inzwischen mit Umber in die Verliese der Burg hinab. Umber zeigte Praxis einen Geheimgang, der zu einer Truhe mit allerlei wertvollen Edelsteinen führte. Eine Mischung aus Feueressenz und der Reagenz ließ einen der Steine in einem besonderen Licht erglühen. Praxis stellte fest, daß es der zweite gesuchte Stein, der Earth Stone, sein mußte. Praxis blieb dann nichts weiter übrig, als auf unsere Ankunft zu warten. Wir holten Praxis mit dem Seil ans Ufer, Umber wollte uns nicht begleiten. Auf unserem weiteren Weg gelangten wir zu einer alten Mühle. Praxis schaute sich den Mechanismus an und fand einen Weg nach unten. Mit seinem glühenden Stab verschafften wir uns gute Sicht, und Praxis stieg hinab. Er ging zunächst nach rechts, wo er eine seltsame Vorrichtung an der Wand entdeckte.



Dungeon Master: Karte Level 4

Eine Untersuchung förderte zwei uhrenartige Scheiben mit Runen zutage. Die Zeiger der Uhren konnte er verstellen. Praxis ging zunächst zurück und gelangte an eine Reihe von Gruben. In der letzten war ein Sonnenstrahl zu sehen. Praxis ging zur ersten flachen Grube zurück und zauberte Wind. Jetzt konnte er alle Runen bis auf die in der letzten Grube erkennen. Am Rad stellte er das linke auf die in der mittleren Grube gefundenen Rune, das rechte auf die andere. Als er fertig war, drückte er auf den Knopf.

In der flachen Grube konnte er nun eine Hacke finden, mit der er Erde und Feueressenzen aus dem Fels hacken konnte. Der Rückweg war ihm aber durch Felsen versperrt. Um Essenzen zu sparen, entschied er sich dafür, sich nicht zu elevieren. Nun hatte er es nicht leicht, aus der Mühle zu entkommen. Es funktionierte aber mit dem Rad (unbedingt speichern!). Er stellte das linke Rad auf die Rune aus der flachen Grube, und am rechten versuchte er eine Stellung nach der anderen, bis folgendes klappte: Er stellte das rechte Rad ein und drückte den Knopf. Dann ging er zu den Gruben, kletterte nach unten und versuchte, dem Licht zu folgen. Klappte es, hatte er es geschafft, wenn nicht, one more try...

Nach diesem Abenteuer gelangten wir automatisch zurück zu Astrix, der uns glücklich empfing. Jetzt müssen wir uns noch auf die Suche nach dem "Anvill", dem letzten Stein, machen, der sich auf der Misty Isle befinden soll. Astrix warnt uns noch vor dem Dread Lord, der das Verschwinden der Steine bemerkt hat und der über die Stadt Zan herrscht. Ich übergebe meine Aufzeichnungen Astrix, damit er sie, falls wir nicht zurückkommen, an die Öffentlichkeit geben kann und andere unsere Mission vervollständigen.

Das ist bislang das einzige, was wir von der kleinen Gruppe in Erfahrung bringen konnten. Vielleicht erfahren wir in einer der nächsten Ausgaben, wie es ihnen auf der Suche nach dem Anvill ergangen ist.

Den Spielständen an den Kragen

Viele Spiele sind sehr komplex, und es ist deswegen gerade zu Beginn sehr schwer voranzukommen. Wir wollen Ihnen im folgenden zeigen, daß und wie es möglich ist, gespeicherte Spielstände von Programmen zum eigenen Vorteil zu verändern. Die Werkzeuge, die man dazu benötigt, sind denkbar einfach. Es reicht ein Diskettenmonitor, der entweder Track- oder File-orientiert arbeitet. Solche Programme gibt es in PD zuhauf, so daß eine Anschaffung den Geldbeutel nicht zu sehr belastet. Die Spiele, deren Dateien verändert werden sollen, sollten allerdings auch einige Voraussetzungen erfüllen:

1. Die Original-, zumindest aber die Spielstanddiskette sollte kopierbar sein, damit nicht das wertvolle Original zerstört wird.
2. Die Spielstände sollten in einem vom Amiga DOS lesbaren Format abgespeichert sein.

Sämtliche Arbeiten an den Spielständen sollten wirklich nur auf einer Kopie erfolgen, da andernfalls Fehler fatale Folgen nach sich ziehen könnten.

Als erstes muß festgestellt werden, wohin das Programm die Spielstände speichert. Der einfachste Fall liegt vor, wenn es das in eine Datei tun würde. Dann brauchen wir nämlich im Inhaltsverzeichnis nur nach einem entsprechenden File Ausschau zu halten und dieses in unseren File-Monitor zu laden. Andernfalls erweist sich ein Diskettenlaufwerk mit digitaler Track-Anzeige von großem Nutzen, da man verfolgen kann, in welchen Track das Programm seine Save-Daten schreibt.

Sind diese Anfangsschwierigkeiten aus dem Weg geräumt, kommt der schwerste Teil. Ihr Diskettenmonitor dürfte zur Zeit nur ein Wirrwar aus hexadezimalen Zahlen (Zahlensystem mit Ziffern von 0 bis 9 und zusätzlich den Buchstaben von A bis F) und/oder ASCII-Zeichen anzeigen. So sind die Daten vom Programm auf Diskette gespeichert worden. Die Programmierer sind im allgemeinen nicht so freundlich dazuzuschreiben, was die Bytes bedeuten. Um das herauszufinden, gibt es zwei Methoden: Assembler-Programmierer suchen sich jetzt im Hauptprogramm die Speicherroutine und schauen einfach, in welcher Reihenfolge das Programm welche Daten abspeichert.

Da der normale User allerdings nicht zu dieser Kaste gehört, bleibt für ihn nur die Methode "Versuch und Irrtum". Dazu verändern Sie entweder gezielt einzelne Bytes in der Datei und schauen, was sie im Programm bewirkt haben, oder Sie verändern den Spielstand nur um Kleinigkeiten und schauen, was sich verändert hat. Nach soviel Theorie jetzt ein kleines Beispiel.

Schauen wir uns doch einmal das Listing zu Starflight an. Nachdem das Programm die Datei geöffnet hat, schreibt es an verschiedene Stellen den Wert fünf beziehungsweise sechzehn. Das sind die Werte für die optimale Bewaffnung und die sechzehn möglichen Frachtcontainer. Will man das selbst feststellen, kauft man sich für sein Schiff eine gute Bewaffnung, speichert ab und schaut, wie sich das File im Detail verändert hat. Manche Diskettenmonitore haben extra dazu eine Möglichkeit, Files direkt zu vergleichen und die Unterschiede anzuzeigen. Wenn an dieser Stelle statt einer Eins eine Fünf steht und ansonsten alles gleichgeblieben ist, kann man sicher sein, daß man mit den Werten von eins bis fünf die Stärke der Bewaffnung einstellen kann. Bei Starflight kommt noch hinzu, daß für jeden Teil auch die hundertprozentige Einsatzbereitschaft gegeben werden muß, da er sonst vom Spiel als beschädigt oder gar zerstört gewertet wird.

Sie sehen, das Verändern von Dateien zu seinem eigenen Nutzen ist zwar arbeitsaufwendig, aber nicht so schwer. Und hat man erst einmal herausgefunden, welche Bytes was bewirken, kann man auch ein kleines Programm in BASIC oder einer anderen Sprache schreiben, um diese Informationen der Nachwelt zu erhalten oder über eine Zeitschrift zugänglich zu machen. Zum Schnuppern und Ausprobieren finden Sie in dieser Helpline neben dem schon erwähnten Starflight-Patch noch zwei weitere Patches zu "Keef the Thief" und "Demons Winter". Übrigens sollten die drei Programme problemlos in GFA-Basic umzusetzen sein, sofern die spezifischen Eigenheiten dieser Programmiersprache berücksichtigt werden. Vielen Dank auch an Ralf Höhner für die drei Programme.

(Robert Marz/hs)



```
'        Tuning fuer Demon's winter Charaktere
' Das unten stehende Programm erhoeht die Experience Points aller 5 Charktere
' auf ueber 16 Millionen Punkte
' Das verfuegbare Kleingeld wird ebenfalls auf ueber 16 Millionen erhoeht
' Die Jungs freuen sich ueber den Lottogewinn (Ehrlich)
' Damit bestehen keine Probleme mehr beim einkaufen oder ?
' Fuer die Lebensmittel besteht allerdings eine Beschraenkung auf 255 Einheit
' also zwischendurch nachkaufen oder das programm laufen lassen
' Demon's Winter Diskette Kopieren
' Amigabasic laden und starten
' Tuning programm laden
' Demon's Winter Kopie in Laufwerk df0: einlegen
' Tuning programm starten
' der Rest geschieht automatisch
' Es macht mehr Spass das Spiel zu l-sen, wenn man nur das Kapital
' und die Nahrung erh-ht. Fuer charaktere ueber level 20 fehlen die
' Gegner. Die Kaempfe werden langweilig wenn man immer gewinnt.
' Die Superwaffe ist ein unzerstoerbares Amulett mit unendlich vielen
```

Listing: Spieletuning

```
' Ladungen, welches einen 255 Punkte starken Feuersturm erzeugt.
' Das ist absolut toedlich fuer alle Wesen im Spiel - auch fuer sich
' selbst !!!!!!
' Ralf Hoehner
CLEAR
hauptmenu:
CLS
LOCATE 3,6:PRINT"Bitte Demon's Winter Kopie in df0: einlegen"
LOCATE 5,8:PRINT"Nach dem Kopieren mit der Workbench"
LOCATE 6,11:PRINT"Disknamen in  DEMON aendern !!"
LOCATE 10,10:PRINT"1 = experience points   aendern"
LOCATE 11,10:PRINT"2 = taschengeld         aendern"
LOCATE 12,10:PRINT"3 = nahrung             aendern"
LOCATE 13,10:PRINT"4 = skills              aendern"
LOCATE 14,10:PRINT"5 = hit + spell points  aendern"
LOCATE 15,10:PRINT"6 = level               aendern"
LOCATE 16,10:PRINT"7 = speed,strenght etc. aendern"
LOCATE 17,10:PRINT"8 = superwaffe fuer char1 "
```



Listing: Spieletuning

```
LOCATE 18,10:PRINT"9 = ende"
LOCATE 19,10:INPUT"Was soll geaendert werden ?",a
IF a<0 AND a>9 THEN GOTO hauptmenu
 ON a GOSUB expe,geld,essen,skills,hpsp,level,staerke,waffe,ende
GOTO hauptmenu
'*********************************
expe:
OPEN "r",1,"df0:dem_DATA/party.dat",1
FIELD #1,1 AS a$
LSET a$=CHR$(255)  'wert der dem char eingetragen wird
PUT #1,198 'exp points char. 1
PUT #1,199 ' auf 16 millionen
'****************
PUT #1,458 'exp points char. 2
PUT #1,459 ' auf 16 millionen
'****************
PUT #1,718 'exp points char. 3
PUT #1,719 ' auf 16 millionen
'****************
PUT #1,978 'exp points char. 4
PUT #1,979  ' auf 16 millionen
'****************
PUT #1,1238 'exp points char. 5
PUT #1,1239 ' auf 16 millionen
CLOSE #1
RETURN
'****************
geld:
OPEN "r",1,"df0:dem_DATA/party.dat",1
FIELD #1,1 AS a$
LSET a$=CHR$(255)  'wert der dem char eingetragen wird
PUT #1,1312 'geld auf
PUT #1,1313 '16 millionen
PUT #1,1314 'erh-hen
CLOSE #1
RETURN
'****************
essen:
OPEN "r",1,"df0:dem_DATA/party.dat",1
FIELD #1,1 AS a$
LSET a$=CHR$(255)  'wert der dem char eingetragen wird
PUT #1,1456 ' nahrung auf maximum 255 einheiten setzen
CLOSE #1
RETURN
'****************
skills:
OPEN "r",1,"df0:dem_DATA/party.dat",1
FIELD #1,1 AS a$
LSET a$=CHR$(1) ' 1 = skill-kenntnis, 0 = keine kenntnis
    'max. skills fuer charakter 32 skills
FOR i=201 TO 232  'char 1
PUT #1,i
NEXT i
'******************
    'max. skills fuer charakter
FOR i=461 TO 492  'char2
PUT #1,i
NEXT i
'********************
   'max. skills fuer charakter
FOR i=721 TO 752  'char3
PUT #1,i
NEXT i
'*********************
   'max. skills fuer charakter
FOR i=981 TO 1012 'char4
PUT #1,i
NEXT i
'*********************
    'max. skills fuer charakter
FOR i=1241 TO 1272'char5
PUT #1,i
NEXT i
CLOSE #1
RETURN
'***********************
hpsp:
OPEN "r",1,"df0:dem_DATA/party.dat",1
FIELD #1,1 AS a$
LSET a$=CHR$(255) 'wert der dem char eingetragen wird
FOR i = 253 TO 256'max. hp + sp char1
PUT #1,i
NEXT i
FOR i = 513 TO 516'max. hp + sp char2
PUT #1,i
NEXT i
FOR i = 773 TO 776'max. hp + sp  char3
PUT #1,i
NEXT i
FOR i = 1033 TO 1036'max. hp + sp char4
PUT #1,i
NEXT i
FOR i = 1293 TO 1296'max. hp + sp char5
PUT #1,i
NEXT i
CLOSE #1
RETURN
'************************
level:
OPEN "r",1,"df0:dem_DATA/party.dat",1
FIELD #1,1 AS a$
LSET a$=CHR$(255)  'wert der dem char eingetragen wird
PUT #1,245 ' level 255  char1
PUT #1,505 ' level 255  char2
PUT #1,765 ' level 255  char3
PUT #1,1025 'level 255  char4
PUT #1,1285 'level 255  char5
CLOSE #1
RETURN
'***************************
staerke:
OPEN "r",1,"df0:dem_DATA/party.dat",1
FIELD #1,1 AS a$
LSET a$=CHR$(255)  'wert der dem char eingetragen wird
PUT #1,1285 ' level 255
FOR i = 248 TO 252 'max. staerke usw.  char1
PUT #1,i
NEXT i
FOR i = 508 TO 512 'max. staerke usw.  char2
PUT #1,i
NEXT i
FOR i = 768 TO 772 'max. staerke usw.  char3
PUT #1,i
NEXT i
FOR i = 1028 TO 1032 'max. staerke usw. char4
PUT #1,i
NEXT i
FOR i = 1288 TO 1293 'max. staerke usw. char5
PUT #1,i
```



Listing: Spieletuning

```
NEXT i
CLOSE #1
RETURN
'****************
waffe:
OPEN "r",1,"df0:dem_data/party.dat",1
FIELD #1,1 AS a$
LSET a$=CHR$(20) ' wert fuer amulett
PUT #1,13
LSET a$=CHR$(100) 'unbegrenzte nutzungen 0% break-chance
PUT #1,18
LSET a$=CHR$(2) ' wert fuer invoked feuersturm
PUT #1,20
LSET a$=CHR$(255) ' staerke des invoked zaubers
PUT #1,21
LSET a$=chhr$(1) ' gegenstand ist identifiziert
PUT #1,29
CLOSE #1
RETURN
'********************
ende:
CLOSE
SYSTEM
```

```
'       Tuning fuer Keef the Thief
' Das Programm aendert alle Charakterwerte auf 100%  ( max. 255%)
' und erhoeht das Kapital ein wenig. Das Kapital wird nur auf
' ca. 28000 erhoeht. Bei zu hohen Werten hat man minus Betraege
' auf seinem Konto. Das absolute Maximum habe ich allerdings
' nicht ermittelt. Falls Ihr mehr braucht, alles einkaufen was
' Ihr wollt und das Programm nochmal starten.
' Die Kaempfe werdet ihr auch mit nur 100% auf jeden Fall gewinnen
' das sollte genuegen ohne den ganzen Spielspass zu nehmen
' Ralf Hoehner
CLEAR
CLS
LOCATE 3,6:PRINT"Bitte Keef Kopie in df0: einlegen"
LOCATE 5,8:PRINT"Nach dem Kopieren mit der Workbench"
LOCATE 6,11:PRINT"Disknamen in keef aendern !!"
CLEAR
LOCATE 10,10:PRINT"Bitte Diskette in Df0: einlegen"
LOCATE   11,10:PRINT   "Taste druecken"
WHILE  INKEY$=""

WEND

'****************
OPEN "r",1,"df0:sg",i 'spielstand-datei oeffnen
FIELD #1,1 AS a$
'****************              save
LSET a$=CHR$(100)' % zahl der staerke usw. wehr mehr will 255 eintragen
PUT #1,3 'werte eintragen
PUT #1,5 'fuer strenght usw.
PUT #1,7
PUT #1,9
PUT #1,11
PUT #1,13
PUT #1,15
PUT #1,17
PUT #1,19
PUT #1,23
PUT #1,29
'****************
' das kapital auf ca. 29000 erhoehen
LSET a$=CHR$(111)
PUT #1,26
LSET a$=CHR$(255)
PUT #1,27
'****************
PRINT " Fertig "
```

```
 '       Tuning fuer Starflight
' Das Programm aendert nur fuer Spielstand 1 die Ausruestung
' des Raumschiffes. Falls Ihr am Anfang mehr Geld braucht,
' solltet Ihr die gesamte Ausruestung verkaufen und das
' Programm nochmal durchlaufen lassen , allerdings macht das
' Geldverdienen durch Mineralien sammeln mehr Spass, deswegen
' gibt zumindest von mir keine Mogeltips fuer andere
' Gegenstaende. Ich meine, dass eine optimale Bewaffnung etc.
' genuegt.
' Ralf Hoehner
CLEAR
CLS
LOCATE 3,6
PRINT"Bitte Starflight Kopie in df0: einlegen"
LOCATE 5,8
PRINT"Nach dem Kopieren mit der Workbench"
LOCATE 6,11
PRINT"Disknamen in Starflight aendern !!"
LOCATE 11,10
PRINT "Taste Druecken"
WHILE
 INKEY$ =""
WEND

OPEN "r",1,"df0:game.dat",1
FIELD #1,1 AS a$
'****************
LSET a$=CHR$(5)
PUT #1,4
' Werte fuer optimalen Laser etc. eintragen
PUT #1,6
PUT #1,8
PUT #1,10
PUT #1,12
'****************
LSET a$=CHR$(16)
PUT #1,14
' Wert fuer 16 Frachtcontainer
'****************
LSET a$=CHR$(100)
PUT #1,5
'alle Teile zu 100% einsatzbereit
PUT #1,7
PUT #1,9
PUT #1,11
PUT #1,13
PUT #1,15
'****************
PRINT " Fertig "
```

Listing: Spieletuning



Schatzsuche im Wrack der gesunkenen Esmeralda



Als Doppeldeckerpilot auf den Spuren des Roten Barons

Demnächst auf Ihrem Computer

Schätze, Flieger und Abenteuer: Die Computerspielindustrie braucht sich über mangelnde Fantasie und Kreativität nicht zu beschweren. Auch wenn wirklich neue Spielkonzepte rar sind, zeigt sich, daß durch Variation, gute Grafik und Sound noch viel Sehens- und Spielenswertes realisiert werden kann.

Treasure Trap

Ein Wrack voller Goldbarren ist der Schauplatz dieses neuen Jump'n'Run-Spiels aus England. Das Spiel stellt sich im "Puppenstuben-Look" dar. Die Demo, die wir davon in Augenschein nehmen konnten, sah durch viele nette Grafiken und Animationen recht vielversprechend aus. Um an die Schätze der Esmeralda heranzukommen, muß der Spieler Köpfchen beweisen. Vielfach gilt es, sich umherirrender Tiere zu bedienen, um von ihnen Barren aus unzugänglichen Nischen schieben zu lassen.

Wings

Die Luftkämpfe des ersten Weltkriegs stellen den Hintergrund dieses neuen Cinemaware-Spiels, in dessen Verlauf Sie sogar dem legendären Roten Baron über den Weg fliegen können. Wings fügt sich nahtlos in die Reihen der anderen Cinemaware-Titel ein, und der Spieler findet wie gewohnt gute Grafik und feinen Sound. Jedoch zeigt sich auch in der Spielhandlung von Wings wieder die moralische Unbeschwertheit der Software-Schaffenden in den USA.

Final Battle

Textadventures haben es nicht leicht am Markt; aus dem zweiten Teil von "Legend of the Sword" wird deshalb ein Icon-Spiel mit Rollenspielelementen werden. Das Programm steht noch in der Entwicklung, aber es ist uns gelungen, ein Bildschirmfoto (aus einem sehr frühen Stadium) zu ergattern. Von der eigentlichen Spielhandlung stand fest, daß es ein Wiedersehen mit allen bekannten und beliebten Fieslingen aus dem ersten Teil geben wird.

Up & Away

Auf einem fernen Planeten herrscht ein eklatanter Wassernotstand. Also werden zwei Spezialisten auf einen anderen Stern geschickt, auf dem es reichlich H_2O gibt. Dort gilt es, das feuchte Kleinod einzusammeln und auf den eigenen, darbenden Planeten "zurückzubeamen". Und so muß man Schauplatz für Schauplatz nach Feuchtigkeit durchsuchen. Hier stellt sich ein Geschicklichkeitsspiel mit Splitscreen vor, das insbesondere zu zweit viel Spaß macht.

(hs)



Final Battle wird komplett über Icons gesteuert



Auf Wassersuche im Amazonasdschungel: Up and Away

Marshal M. Rosenthal

Vorsicht, Abkürzung

Talking Handhelds

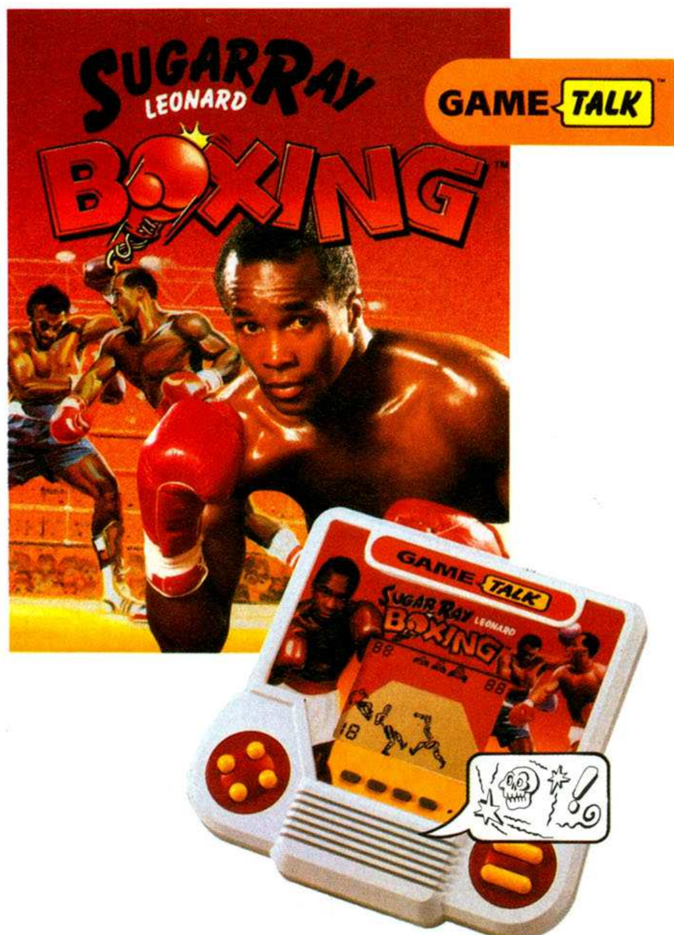
Viele Videospiele, die Sie aus den Arcaden kennen, werden nach einer gewissen Zeit auch auf Heimcomputer oder Spielkonsolen umgesetzt. Neu im Trend sind jedoch die sogenannten Handhelds, kleine netzunabhängige Videospiele mit LCD-Schirm. Brandneu sind die Talking Handhelds, die Sie mit etwas Glück bei uns gewinnen können.



In Zusammenarbeit mit Tiger Electronics und Marshal M. Rosenthal, den Sie bereits aus unseren Berichten über das Geschehen in den USA kennen, haben wir diesmal einen ganz besonderen Wettbewerb ausgearbeitet.

Viele interessante elektronische Spielereien werden zuerst und meist auch ausschließlich in den USA vorgestellt. Eigentlich schade, wenn man bedenkt, daß auch hier in Deutschland ein reges Interesse daran besteht. Diesmal können wir nicht nur mit Informationen aufwarten, Sie können auch die sprechenden Handheld-Games gewinnen! Tiger Electronics ist ja für ihre Unzahl an LCD-Handheld-Games bekannt.

Sicher haben Sie auch schon einmal vom Gameboy und dem Lynx-System gehört. Nur, keines dieser Systeme spricht. Jedes der Spiele, die wir hier ausgeschrieben haben, verfügt über eine lebensechte Synthesizerstimme, die den Spieler durch jede Phase des Spiels führt, Ratschläge gibt, vor Gefahren warnt und wertvolle Zusatzinformationen erteilt. Unter den neuen Spielen findet man zwei Sportspiele, »Sugar Ray Leonard's Talking Boxing« und »All Star Baseball«. »Snake's Revenge« ist ein Rambo-ähnliches Actionspiel, in dem es darum geht, einen (fast) unbesiegbaren Nuklearpanzer in die ewigen Jagdgründe zu schicken.



»Ninja Gaiden II« schließlich entführt Sie in eine Welt voller Ninjas der übelsten Sorte, die zu allem Überfluß auch noch ihre Freunde mitgebracht haben. Ninja Gaiden II ist der Nachfoger eines bekannten Spiels aus den 80ern. Es enthält 20 Levels, 10 Szenen und 15 Items, mit denen man die eigene Kraft erhöhen und spezielle Fähigkeiten erlangen kann.

Die Aufgabe:

Stellen Sie sich vor, der Begriff **AMIGA** wäre eine Abkürzung. Was könnte sich hinter diesen fünf Buchstaben verbergen? Wir erwarten Ihre möglichst lustige und originelle Interpretation dieser Abkürzung.

Die Teilnahmebedingungen:

○ Schreiben Sie Ihre Interpretation von **AMIGA** auf eine Postkarte.

○ Senden Sie Ihre Zuschrift an den

DMV-Verlag
Redaktion AMIGA DOS
Stichwort: Talking Handhelds
Postfach 250
3440 Eschwege

Und bitte den Absender nicht vergessen!

○ Mitarbeiter des DMV-Verlages und deren Angehörige sowie Mitarbeiter von Tiger Electronics dürfen leider nicht teilnehmen.

○ Alle Einsendungen nehmen an der Verlosung teil.

○ Einsendeschluß ist der

08.08.1990

○ Der Rechtsweg ist ausgeschlossen.

(mm)



Liebe Leser,
in dieser Rubrik finden Sie neben Informationen zu den in der AMIGA DOS vorgestellten Programmen und Produkten Rat und Hilfestellung zu Ihren kleineren und größeren Programmierproblemen. Wir sind natürlich jederzeit bemüht, die eingehenden Leserfragen zu beantworten. Doch haben Sie bitte Verständnis, daß wir nicht alle eingehenden Briefe persönlich beantworten können. Oft erreichen uns mehrere Briefe zum gleichen Thema, einer davon wird dann stellvertretend für die anderen in unserer Zeitschrift beantwortet.
Dringende Probleme, die das Heft betreffen, lassen sich möglicherweise besser telefonisch regeln. Rufen Sie dienstags unsere Hotline an. Von 17 bis 20 Uhr stehen wir Ihnen mit Rat und Tat zur Seite, wenn Sie eine der folgenden Nummern wählen: 0 56 51/8 09-7 40 (bis 744)
Ihre AMIGA-DOS-Redaktion

SIE FRAGEN, WIR ANTWORTEN

Hier nun wieder unsere Leserbriefecke, in der Sie Fragen stellen oder beantworten können. Wenn Sie uns schreiben wollen, richten Sie Ihr Schreiben an den

DMV-Verlag,
Redaktion AMIGA DOS,
Leserservice,
Postfach 250,
3440 Eschwege.

Wenn Sie eine Antwort für die Problemecke haben, schicken Sie diese unter dem Kennwort »Problemecke« ein.

Programm "Taigong" aus Ausgabe 6/90

Voll Freude entdeckte ich den Abdruck meines Listings "Taigong". Leider mußte ich nach der Durchsicht einen kleinen Fehler im Listing feststellen, der sich aber leicht beheben läßt. Der Fehler befindet sich in der Zeile 215. Aus der dort abgedruckten Zeile

»`IF name$<>"" THEN...`« *muß die Zeile in* »`IF nam$<>""`«

geändert werden.

Stefan Buschmann, Frankfurt

Listings zu kaufen?

Ich sehe immer viele Programme zum Abtippen in Ihren Heften. Doch das Abtippen ist nicht gerade meine Stärke, deswegen möchte wissen, ob es die Programme auch zu kaufen gibt? Wenn ja, wie geht das?

Mark Spitzer, Berlin

Natürlich sind die abgedruckten Listings auf Disketten erhältlich. Welche Programme sich auf der sogenannten Databox befinden, können Sie anhand des Diskettensymbols im Listing erkennen.
Nehmen Sie einfach die Bestellkarte aus der jeweiligen Ausgabe heraus und markieren die gewünschten Disketten.

(Red.)

»Deluxe-Paint« und Genlock, die zweite

Sie haben in den letzten Ausgaben der Amiga DOS einige Tips zu Genlock in Verbindung mit »D-Paint« gegeben. Die Empfehlungen waren auch bis auf ein paar Einschränkungen gut. »D-Paint« verwendet nicht den Workbench-Maus-Pointer, sondern ein Fadenkreuz, welches sich nicht als Pfeil eignet. Nach einigen Versuchen habe ich eine Lösung gefunden. Man sollte wie folgt vorgehen:

- *»Deluxe-Paint« laden und ein Fenster, zum Beispiel »LowRes Overscan«, öffnen,*
- *mit der Funktionstaste [F10] den Rahmen ausblenden,*
- *mit der linken Amiga-Taste und der Taste [N] (beide gleichzeitig drücken) zur Workbench-Ebene zurückgehen,*
- *»Preferences« aufrufen,*
- *Edit-Pointer anwählen,*
- *Pointer nach Wunsch editieren,*
- *»OK« anklicken,*
- *»USE« anklicken,*
- *Mit der linken Amigataste und der Taste [M] auf den D-Paint-Screen wechseln, und schon hat man seinen Zeigepfeil für das Genlock.*

Man darf jetzt auf keinen Fall eine der beiden Maustasten bedienen, da der Pointer sonst wieder zum D-Paint-Fadenkreuz wechselt. Hat man versehentlich eine Maustaste gedrückt, so muß man mit der Tastenkombination linke Amigataste plus [N] zur Worbench zurückkehren und diese einmal anklicken. Der Pointer wechselt wieder und mit der Kombination linke Amigataste plus [M] kann man nun zu »D-Paint« wechseln.

Auf meinem Amiga 1000 (die gibt es noch) läuft dieses Verfahren unter Kickstart 1.3 und »Deluxe-Paint III« mit einem Electronic-Design-Genlock problemlos. Es dürfte auch mit »D-Paint II« und Workbench 1.2 und einem anderen Genlock funktionieren.

Und nun etwas anderes:

Oft ist es nicht bekannt, was für Dateien und Programme sich auf einer Public-Domain-Diskette befinden. Mit folgender Befehlsfolge im CLI kann man das gesamte Inhaltsverzeichnis einer Diskette ausdrucken lassen (Workbench in »DF0:«, PD-Diskette in »DF1:«)

```
DIR >prt: DF1: opt a
```

Auch einzelne Textdateien lassen sich so ausdrucken.

```
TYPE >prt: DF1:Dateiname
```

Übrigens betreibe ich einen Userclub in Süddeutschland, der sich aber nicht nur auf diesem Raum beschränkt.
Infos:
Axel Schubert
Stichwort:Userclub
Sachsenheimer Str. 5
7141 Oberriexingen

Antworten zu H. P. Pesch und M. Wellmann, aus Heft 3/90

Nach dem Durchlesen der Amiga DOS sind mir zwei Leserbriefe aufgefallen.
Zu dem Brief von H. P. Pesch:
Ich würde das Programm »3rd Day« (Cactus 34) empfehlen. Es findet fast alle Bilder, die nach einem Reset noch im Speicher bleiben. Da die erwähnte Diskette Public Domain ist, kann ich mir nicht vorstellen, daß beim Reset der Speicher gelöscht wird, wie es bei den meisten Spielen der Fall ist.
Zu dem Brief von M. Wellmann:
Die "Erscheinungen" können an einem defekten Monitorka-

bel liegen. Bei einigen Bekannten, deren Monitor plötzlich grün, gelb gestreift und so weiter wurde, half ein heftiger Ruck am Monitorkabel, und das Problem war gelöst.
Und hier noch ein kleiner Tip für defekte interne Amiga-Laufwerke: einmal kräftig draufhauen.

Hajo Noerenberg, Dormagen

Zu den Tips: Grundsätzlich kann man sagen, daß Programme nach einem Reset aus dem Speicher entfernt werden. Aber in einigen Fällen bleibt sogenannter Datenmüll im Speicher zurück, der auch nicht ohne weiteres gelöscht werden kann. Abhilfe schafft nur ein Kaltstart (Rechner aus- und wieder einschalten). Ein kräftiger Ruck am Monitorkabel ist bei einem Scart-Anschluß nicht besonders empfehlenswert, da die Befestigung wirklich nicht umwerfend ist. Aber ansonsten ist der Tip in manchen Fällen hilfreich.
Daß die "Fernsehermethode" auch bei den Amiga-Laufwerken angewandt werden kann, ist auch uns nicht bekannt gewesen, aber einmal ist immer das erste Mal. Wir weisen allerdings darauf hin, daß man auch bei solchen Brachialmethoden immer mit Verstand arbeiten sollte.

Probleme mit dem A2000 und dem Programm »X-Copy«

Als Besitzer eines Amiga 2000 (Version C) kaufte ich mir kürzlich das Kopierprogramm »X-Copy II« Version 2.1. In der Grundeinstellung des Programms stehen mir als Benutzer 809265 Bytes zur Verfügung. Rufe ich jedoch die Option »Kill-Sys« auf, so habe ich anstatt über einer Million jetzt nur noch 478720 Bytes zur Verfügung. Ich habe das Programm daraufhin noch mehrmals gestartet, kam aber immer wieder zum selben Ergebnis.
Daraufhin habe ich das Programm auf einem Amiga 2000B (jedoch mit zusätzlich eingebautem Fat AGNUS) getestet. Ich glaubte, auf das gleiche Ergebnis zu kommen, da durch den zusätzlichen Einbau eines Fat AGNUS in den A2000B die zwei Computer annähernd gleich sein müßten. Zu meiner Überraschung mußte ich feststellen, daß mir hier jedoch wie nach Vorschrift über eine Million Bytes zur Verfügung standen. Ich kann mir dieses Phänomen nur dadurch erklären, daß entweder mein Computer kaputt ist oder daß das Programm mit der C-Version des Amiga nicht klarkommt. Die erste Alternative ist jedoch weitgehend auszuschließen, da mir im normalen Betrieb (zum Beispiel Workbench) fast ein Megabyte zur Verfügung steht.

Dieter Miess, Hallstadt

Eine Anfrage bei der Firma Cachet ergab folgendes: Alte Versionen von X-Copy erkennen nicht die neuen Megabyte-Chips, wird aber in einer neuen Version geändert. Registrierte Kunden bekommen ein »Up-Date« zur Verfügung gestellt.

(Red.)

Amiga 500 und der Fat AGNUS

Ich besitze einen Amiga 500 mit Kickstart 1.3, in den der neue Fat AGNUS schon werksseitig eingebaut war. Die Aufrüstung war jedoch noch nicht vorgenommen worden.
Soviel vorweg: Ich habe die Umrüstung durchgeführt und keine größeren Unregelmäßigkeiten seit der Änderung festgestellt. Nur bei einem Guru verhält sich der Amiga etwas anders als vorher.
Ich habe mit dem ganzen nur ein Problem: Ich bin nicht sicher, ob ich alles richtig gemacht habe. Die Umrüstung entnahm ich nach einer Anleitung einer anderen Amiga-Zeitschrift. Laut dieser waren zwei Dinge nötig:
1. *Jumper JP2 durchtrennen und neu zusammenlöten,*
2. *eine Leiterbahn in der Nähe des Memory-Schachtes durchtrennen.*

Eben diese Leiterbahn ist aber auf meinem Board nicht vorhanden. Besser gesagt, gibt es an etwa der gleichen Stelle einen Jumper JP7, der zum gleichen Pin am Memory-Schacht führt wie besagte Leiterbahn in der Anleitung. Also habe ich risikofreudig diesen Jumper durchtrennt und – wohlgemerkt – nicht neu zusammengelötet. Und siehe da, es klappte. Ich war mir aber nicht sicher und versuchte es sowohl mit der einen als auch mit der anderen Jumperstellung ebenfalls. In beiden Fällen weigerte sich der Computer zu starten. Nur mit vollkommen durchtrenntem Jumper JP7 funktionierte es!
Nun meine Frage: Habe ich alles richtig gemacht, oder muß JP7 in einer von beiden Stellungen stehen und eventuell ein anderer Jumper durchtrennt werden?
Können mit meiner Vorgehensweise später irgendwelche Inkompatibilitäten oder gar Schäden an künftigen Hardware-Erweiterungen entstehen? Ich denke zum Beispiel an eine interne 2,5 Megabyte-Erweiterung.

Norbert Bendl, Nersingen

Sie haben alles richtig gemacht, denn sonst wäre der Amiga nicht mehr am "Leben". Wir wissen zwar nicht, von welcher Version die freundliche Amiga-Zeitschrift ausgegangen ist, aber durch die vielen Amiga-Versionen kann schon einiges durcheinandergeraten. Wenn Ihr Amiga mit dem durchtrennten Jumper läuft, dann sollten Sie auch nichts mehr daran ändern.
Und irgendwelche Inkompatibilitäten brauchen Sie auch nicht zu einem späteren Zeitpunkt zu befürchten.

(Red.)

Virus über Btx?

Da mich das Btx-Fieber gepackt hat, möchte ich Ihnen ein paar Zeilen schreiben.
Zunächst stand ich vor dem Problem, welchen Btx-Dekoder ich mir zulegen sollte. Nachdem ich einige Tests in verschiedenen Computer-Zeitschriften gelesen hatte, fiel meine Wahl auf Multiterm-pro. Ich verwende den Dekoder mit der D-BT03 Postbox.
Es macht Riesenspaß, mit den Programmmen zu arbeiten beziehungsweise zu Btx-en. Erwähnenswert bei Muliterm-pro ist die Möglichkeit, Btx-Sitzungen zu protokollieren. Da wird alles mitgeschnitten, und man kann hinterher offline (ohne weitere Kosten) alles noch einmal durcharbeiten und ausdrucken.
Der Telesoftware-Dienst ist auch eine prima Sache, man kann sich von dort aus ein Programm auf seinen Amiga rüberziehen. Allerdings habe ich mich bis jetzt davor gescheut. Jetzt hätte ich in diesem Zusammenhang eine Frage: Ist es möglich, sich einen Virus einzufangen, wenn Btx-Software überspielt wird?
Noch eine Frage: Wann werde ich Sie im Btx-Anbieter-Verzeichnis finden?

Martin Wünn, Stuttgart

Grundsätzlich können Sie sich auch über Btx einen Virus einfangen, der sich dann im System breitmacht, aber jeder Anbieter von Amiga-Btx-Software wird darauf achten, daß sich kein Virus im Programm befindet.
Zur zweiten Frage müssen wir Ihnen leider sagen, daß wir bis jetzt noch keine Zeit hatten, diesen Service einzurichten. Sollte aber solch ein Projekt von uns in die Wege geleitet werden, dann werden wir die Leser rechtzeitig unterrichten.

(Red.)

Der Amiga hat das Zeitliche gesegnet

Während ich ein Assemblerprogramm auf meinem Amiga laufen ließ und zwischendurch an einer kleinen Lichtorgel gebastelt habe, passierte es. Die Sicherung flog raus. Nach dem Einschalten sprang alles außer meinem Amiga wieder an (das Netzteil war noch eingeschaltet). Er weilte nicht mehr unter uns. Einige Sachen hatte ich getestet, am Parallelport liegt keine 5V-Spannung an, auf einen Warmstart (Tastatur-Reset) reagiert er noch, das zeigt sich durch kurzes Aufleuchten der Caps-Lock-LED.
Am Netzteil ist auch alles in Ordung, es liefert die gewünschte Spannung, doch das "Klicken" des Laufwerks fehlt. Nach dem Einschalten gibt es immer einen grünen Bildschirm und anschließend hängt sich der Rechner auf. Was kann das bloß sein?

Nach unserer Meinung ist entweder der 8520-Chip, PAULA oder GARRY defekt. Dies läßt sich aber nur bei einem Fachhändler überprüfen. Bitte wenden Sie sich deshalb an ihn.

(Red.)

Neuer Virus?

Ich habe seit Monaten ein Problem, mein Rechner Amiga 500 stürzt dauernd ab. Jetzt habe ich auf einmal einen Requester nach einem Absturz, der folgerdermaßen aussieht:

Die Problemecke

Wir haben wieder Briefe gesammelt, die wir in der Redaktion nicht beantworten konnten. Sollten Sie, liebe Leser, etwas zu diesen Themen wissen, dann schreiben Sie uns bitte unter dem Kennwort "Problemecke". Sie helfen vielleicht vielen Amiga-Besitzern.

Abschaltbare Festplatte

Zunächst möchte ich Ihnen mein Lob über Ihre neue Zeitschrift Amiga DOS ausdrücken. Sie hebt sich angenehm von den anderen Zeitschriften ab und bringt endlich die Artikel für den normalen Amiga-User. Bitte weiter so.

Nun zu meinem Problem: Nach längeren Tests und Preisvergleichen habe ich mich für eine Festplatte von Commodore (A 590) entschieden. Leider mußte ich feststellen, daß ein Diskettenstart von »DF0:« mit längeren Wartezeiten verbunden ist. Auch den Lüftermotor finde ich auf die Dauer nervtötend. Wer kann mir helfen und eine Hardware-Lösung bieten, die den Festplattenbetrieb vom Expansionsport abschaltet. Löterfahrung und etwas Elektronikverständnis sind vorhanden.

Reinhard Baust, Erlangen

Textomat und Drucker

Ich besitze einen 24-Nadel-Drucker Seikosha SL-80 IP. Leider habe ich Probleme, diesen an den Amiga 500 anzupassen. Speziell gilt dies für das Textomat-Programm von Data-Becker.
Nun meine Frage: Können Sie mir die genauen DIP-Schalterstellungen nennen beziehungsweise Anleitungen für die weitere Vorgehensweise geben?

Günther Baierlacher, Erlangen

Tod einer Festplatte

Seit kurzem besitze ich einen Amiga 2000 mit Festplatte, den ich gebraucht gekauft habe.
Auf der Festplatte befand sich ein Programm namens »Diskmaster«, von dem mir leider die Beschreibung fehlt.
Bei der Benutzung des Diskmasters ist mir dann ein Fehler unterlaufen. Als ich mit diesem Programm die Festplatte aufgerufen habe, habe ich wahrscheinlich die Initialisierung zerstört.
Bei Einlegen der Workbench 1.3 kommt der System-Requester mit der Meldung »Softwarefehler« und nach dem Betätigen der linken Maustaste erfolgt die Guru-Meldung »00000003.00C05170«. Die Festplatte ist von Western Digital mit der Nummer WD 362 und der Nummer SN 82400700.

Meine Frage lautet nun: Wie kann ich den Fehler beheben?

Außerdem lese ich in diesem Zusammenhang immer etwas über ein Prep-Programm. Benötige ich dieses dafür?

Uwe Sandtfoß, Hamburg

Amiga und GFA-BASIC

Seit einiger Zeit habe ich ein Problem, das mir einiges Kopfzerbrechen bereitet. Wenn ich unter GFA-BASIC einen Screen öffne, dann stürzt der Interpreter von Zeit zu Zeit beim Schließen des Screens mit Closes mit der Guru Meditation »8100009« (Free Twice) ab. Dieser Effekt tritt auch dann auf, wenn ich mein Programm einem Testlauf unterziehe und anschließend erneut starte. Der Screen ist beim erneuten Start noch offen (ich habe mein Programm im Testlauf unterbrochen), was aber bei AmigaBASIC keinesfalls zum Absturz führen würde. Leider tritt dieser Fehler sowohl beim Interpreter als auch beim Run-Only-Programm auf und das in allen Versionen. Ich benutze übrigens Kick/Workbench 1.3 und einen Amiga 1000 mit 2,5 Megabyte und eine zweite Floppy.
Meine zweite Frage bezieht sich auf den Einsatz eines Flickerfixers. Es gibt zur Zeit zwei Geräte, die hardwaremäßig das Interlace-Flimmern beseitigen: den Flickerfixer aus den USA und Alcomps Grafikkarte Leonardie. Beide werden in den Video-Slot des A2000 gesteckt und sollen mit einem Multisync-Monitor zufriedenstellende Ergebnisse erzielen. Nachdem aber der Flickerfixer ***nur*** *für den A2000 erhältlich und die A500/1000-Version von Leonardie zu teuer ist, stellt sich mir die Frage, ob ich die Karten nicht an meinem A1000 verwenden könnte. Der Video-Slot ist schließlich eine 36polige Buchsenleiste, die beide Audiokanäle und alle Videosignale bereitstellt. Für einen Flickerfixer dürften hier nur die analogen Farb- und die beiden Synchronisationssignale von Relevanz sein. Diese gibt es aber auch am normalen 23poligen RGB-Ausgang der Amiga-Modelle. Das und die Tatsache, daß Alcomps Leonardie auch für den A500/1000 erhältlich ist, lassen vermuten, daß es mit etwas Geschick möglich ist, die Karten an den "kleinen" Amigas zu verwenden. Wissen Sie diesbezüglich näheres?*

Bernher Kao Innsbruck

»In coming special-Message... Your Amiga is infected by DISASTER-Master V2. Probably the best Virus ever created by Mankind.

Left = Continue Right = Self-Destruction«.

Drücke ich jetzt irgendeine Maustaste, so wird der Bildschirm mit bunten und flimmernden Streifen überzogen. Mache ich jetzt einen Reset, wird der Bildschirm schwarz, und es geht nichts mehr. Ich muß also den Computer ausschalten und wieder neu starten.

Dabei besitze ich alle gängigen Virusvernichtungsprogramme, zum Beispiel »Viruscontrol« und »VirusX 4.0«.

Aber alle diese Programme konnten mir auch nicht helfen! Was für ein Virus ist das? Vielleicht haben andere Amiga-Benutzer schon Erfahrung damit und können mir helfen?

Klaus-Jürgen Hampel, Sibbesse

Es scheint so, daß wieder ein findiger Programmierer keine Beachtung gefunden hat und sich auf diesem Wege bemerkbar machen will.
Leider ist uns der Virus nicht bekannt, und wir können daher auch kein Gegenmittel anbieten. Selbst wir in der Redaktion schützen uns mit »VirusX V4.0«.
Kann hier ein Leser weiterhelfen?

(Red.)

Zum Leserbrief aus Heft 4/90, Oliver B.

In der Amiga Dos 4/90 beklagte Herr Oliver B. die mangelnde Rechengenauigkeit des AmigaBASIC. Unter der Zuhilfenahme des folgenden kleinen Programms müßte das Problem beseitigt sein. Das Programm macht von der Möglichkeit Gebrauch, Fließkommazahlen durch Multiplikation mit entsprechenden Zehnerpotenzen in Integerzahlen umzuwandeln.

```
REM Beispielprogramm
x=1000.1
y=1000
GOSUB fp1
z=x-y
GOSUB fp2
PRINT z
END
REM Floatingpoint Routinen
fp1:
v1=LEN (STR$(x))-LEN
(STR$(INT(x)))-1
v2=LEN (STR$(y))-LEN
(STR$(INT(y)))-1
cv=v1
IF v1<v2 THEN cv=v2
cv=10↑cv
x=x*cv
y=y*cv
RETURN
END
fp2:
z=z/cv
RETURN
```

Das Beispielprogramm der Redaktion zum Problem von Hern Oliver B. (siehe Brief) rechnet im Gegensatz zur Erläuterung völlig exakt. Mir scheint eher, daß sich der Verfasser dieses Programms noch nicht eingehend mit der Anweisung »DEFINT« beziehungsweise mit Integerzahlen überhaupt beschäftigt hat.

Ralph Kröpfl, Gaimersheim

Die Redaktion behält sich vor, Leserbriefe zu kürzen.

Impressum

Herausgeber
Christian Widuch

Redaktionsleitung
Stefan Ritter

Chefredaktion
Markus Matejka (mm)

Leitender Redakteur
Jürgen Borngießer (jb)

Redaktion
Heinrich Stiller (hs), Claus Daschner (cd), Vera Brinkmann (vb)

Freie Autoren dieser Ausgabe
Marshal M. Rosenthal, Bernd Rudolf, Robert Marz, Jürgen Seibel, Michael Anton, Andreas Polk, Ingmar Reyer, Timo Siebert, Peter Woop, Thomas Kolbe, Zuheir Urwani, Rolf Wagner, Dirk Rönnau, D. Kunz, Michael Cordes, Franz-Josef Reichert, Arno Wolff, Antje Hink

Redaktionsassistenz
Anke Kerstan (ke), Susanne Eska (es)

Chef vom Dienst
Matthias Bloß

Produktionsleitung
Gerd Köberich

Bereichsleitung
Claudia Ebbrecht (Fotosatz/Lektorat), Margarete Schenk, Helmut Skoupy (Montage/Reprografie)

Layout
Petra Küch, Patricia Reifenhausen

Fotografie
Klaus Jatho

Fotosatz
Gabriela Joseph

Lektorat
Heike Wiegand

Montage/Reprografie
Manuela Eska, Peter Gajewski

Werbegestaltung
Mohamed Hawa

Anzeigenleitung
Wolfgang Brill

Anzeigenverkauf
DMV-Verlagsbüro München
Zaunkönigweg 2c, 8000 München 82
Telefon (0 89) 4 39 10 87, Telefax 0 89/4 39 10 80
Leitung: Britta Fiebig
Anzeigenverkauf: Monika Schöbel, Peter Schätzle, Hannelore Schulzki, Michael Hofmann

Anzeigenpreise
Es gilt die Anzeigenpreisliste Nr. 1 vom 01.05.1990

Anschrift Verlag/Redaktion:
DMV Daten & Medien-Verlag
Widuch GmbH & Co. KG
Fuldaer Straße 6
3440 Eschwege, Telefon (0 56 51) 8 09-0,
Telefax (0 56 51) 8 09-3 33

Vertrieb
Verlagsunion Erich Pabel-Arthur Moewig KG (VPM), Friedrich-Bergius-Straße 20, 6200 Wiesbaden

Druck
Druckerei Jungfer, 3420 Herzberg

Bezugspreise
»AMIGA DOS« erscheint monatlich.
Einzelpreis DM 6,50/sfr. 6,50/öS 52, –

Abonnementpreise
Die Preise verstehen sich grundsätzlich einschließlich Porto und Verpackung.

Inland:
12 Ausgaben: DM 70, –
6 Ausgaben: DM 35, –

Europäisches Ausland:
12 Ausgaben: DM 100, –
6 Ausgaben: DM 50, –

Außereuropäisches Ausland:
12 Ausgaben: DM 120, –
6 Ausgaben: DM 60, –

Bankverbindungen:
Postscheck Frankfurt/M: Kto.-Nr.: 23043-608
Raiffeisenbank Eschwege:
BLZ: 522 603 85, Kto.-Nr.: 245 7008

SPIELEN SIE GERNE?

Lieben Sie es, in finsteren und unheimlichen Gewölben den letzten Geheimnissen auf den Grund zu gehen? Zeichnen Sie gerne Karten von Gewölben und Spiellandschaften?

Dann sind Sie der richtige Partner für uns.

Denn für unsere AMIGA-DOS-Spieletips suchen wir ständig Tips, Tricks, Karten, Cheatmodes und alles andere, was das Spielerherz höher schlagen läßt. Wenn Sie interessante Informationen, Karten oder Lösungsvorschläge für Computerspiele herausgefunden haben, sind Sie unser/e Mann/Frau.
Durch Ihre Information können Sie vielen Spielern weiterhelfen, die ohne Ihre Hilfe unter Umständen monatelang an der gleichen Problematik "festhängen". Oft sind es nur ein Wort oder ein kleiner Hinweis, die dem Spielgeschehen neues Leben einhauchen.

Aus diesen Gründen möchten wir Sie um Ihre Mitarbeit bitten. Wenn Sie also über Spieletips – gleich welcher Art – verfügen, würden wir uns freuen, von Ihnen Post zu bekommen.

Und hier nun die Adresse, an die Sie Ihre Spieletips schicken können:

DMV-Verlag
Redaktion AMIGA DOS
Stichwort: Spieletips
Postfach 250
3440 Eschwege

PS: Unter allen Einsendungen verlosen wir jeden Monat fünf Software-Titel für die besten Spieletips.

Die Inserenten

Im nächsten Heft

■■■ Programme, Utilities und Spiele

Die Schatzkiste Public Domain steht im Vordergrund der nächsten Ausgabe der AMIGA DOS

■■■ CLI-Kurs

Den Amiga über den CLI beherrschen. Wir vermitteln die Grundlagen.



Bekämpfen Sie die bösen Reptilons auf dem Planeten der Robot-Monster

■■■ Medusa – ein geheimnisvoller Name

Atari-Programme auf dem Amiga? Kein Problem mit dem Medusa-ST-Emulator! Wir haben ihn für Sie getestet.

■■■ Faszination Raytracing

In unserer Werkstatt »Reflections« zeigen wir Ihnen, wie man diese Meisterwerke der dreidimensionalen Kunst erstellt.

■■■ Jingle-Maschine

Spielen Sie doch einfach einmal Diskjockey. Mit diesem Programm zum Abtippen lassen sich Super-Soundeffekte erzielen.

■■■ Utilities

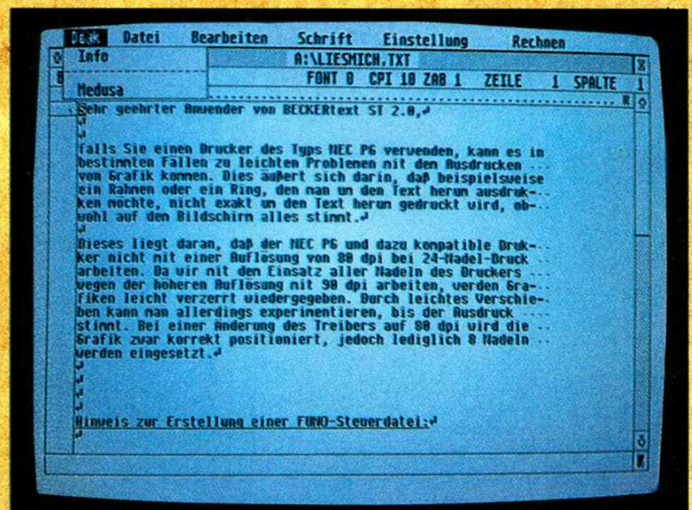
Wir halten wieder eine Vielzahl exzellenter Tips & Tricks zu AmigaBASIC, CLI, Assembler, Modula2, C, GFA-BASIC für Sie parat.

■■■ Spiele und kein Ende

Sagenumwobene Drachen erwarten Sie in dem Spiel »Die Drachen von Laas«. »E« steht für »Einstein«, dem Begründer der Relativitätstheorie wurde ein Computerspiel gewidmet. »E-Motion« ist ein Puzzlespiel der Moleküle. »Escape from the Planet of the Robot Monsters«, so nennt sich ein neues Ballerspiel, das wir Ihnen in der nächsten Ausgabe der AMIGA DOS vorstellen werden.

■■■ Eine geballte Ladung…

Listings, Tests und Infos warten auf Sie.



Im Test: Medusa, der Atari-ST-Emulator

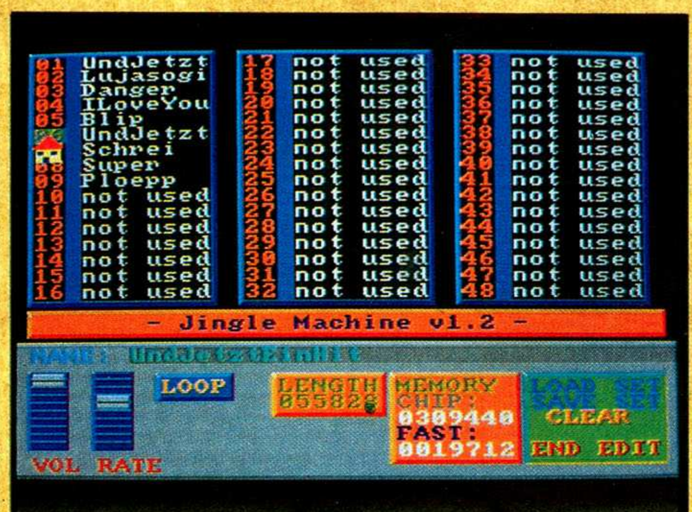
Die nächste



finden Sie ab

08. August '90

bei Ihrem Zeitschriftenhändler

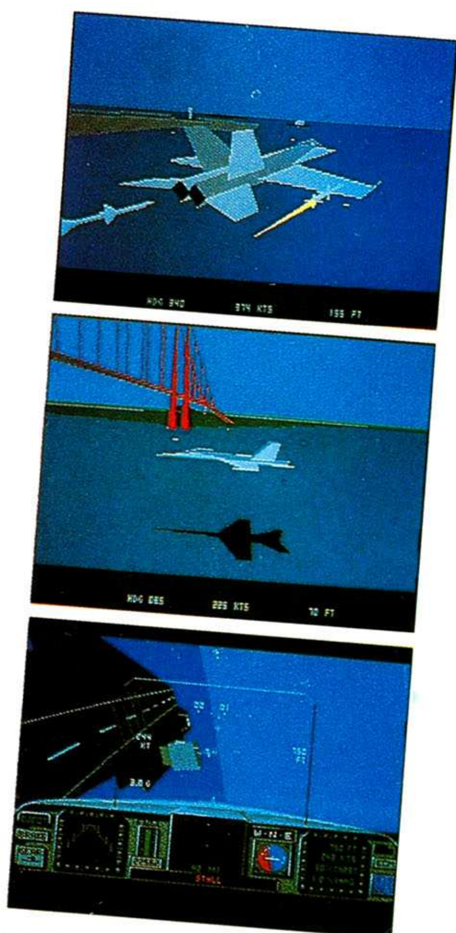


Ploppen, Knacken, Türenknarren, kein Problem für die Jingle-Maschine

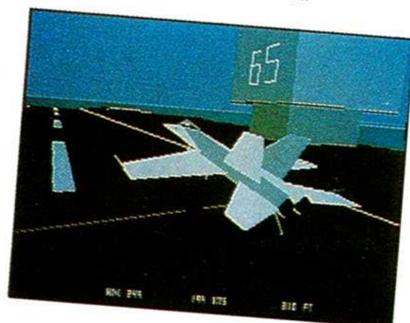
AMIGA DOS

Spielekiste

Solange der Vorrat reicht

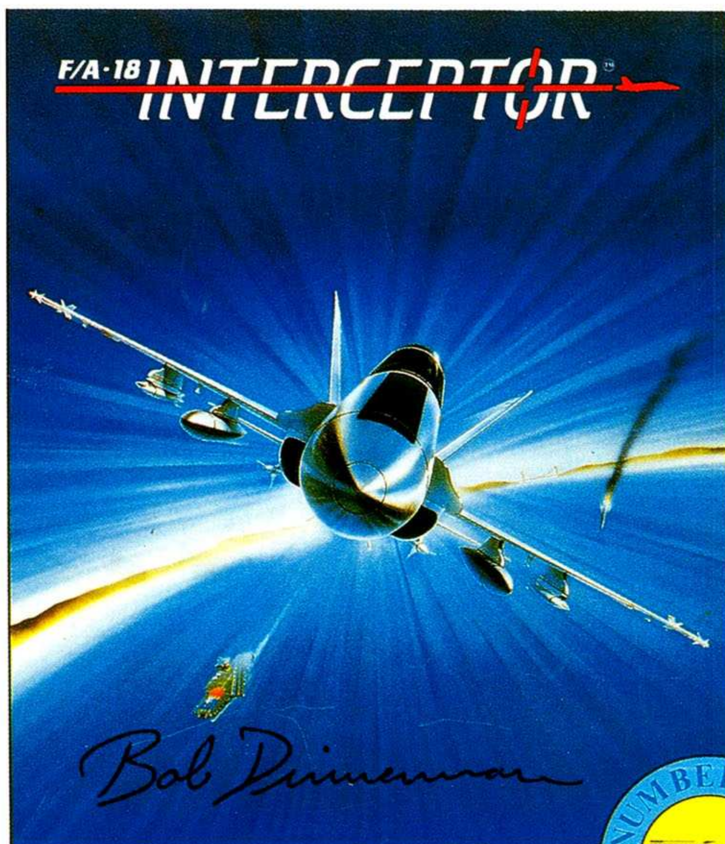


Robert Dinnerman entdeckte seine Leidenschaft für Computerspiele, als er Hardware für Motorola entwickelte. Sein Interesse führte ihn zu der Coin-Op-Abteilung von Bally, wo er an 3D-Perspektiven und Flugsimulationen arbeitete. Nachdem er einen Evans-Sutherland-Flugsimulator gesehen hatte, beschloß Bob, ein vergleichbares, aber erschwingliches Programm für Home-Computer zu entwickeln. Wenn Sie seinen F/A 18 Interceptor spielen, werden Sie sehen, daß er in der Tat etwas Außergewöhnliches vollbracht hat.



Fliegen Sie das faszinierendste Flugzeug unserer Zeit!

Fliegen Sie entweder mit der F16 oder F18, und starten Sie von einem der drei Landstützpunkte oder vom Flugzeugträger aus • Realitätsnahe 3D-Grafik der San-Franzisko-Bucht, komplett mit Golden Gate Bridge, Alcatraz und der Pyramide in San Franzisko • Holografisches HUD-Display und realistische Sound-Effekte • 8 unterschiedliche Ansichten aus dem Cockpit sowie 8 unterschiedliche Außenansichten • 7 Missionen mit unterschiedlichen Schwierigkeitsgraden.



NUMBER ONE IN THE USA · ECA

Unser Sonderangebot an alle Amiga-Leser.
Nur solange der Vorrat reicht.

F/A 18 Interceptor, Electronic Arts,
Amiga 3,5-Zoll-Diskette — DM 39,95*

und aus unserem laufenden Programm

DMV-Fraktalgenerator 3D Amiga
Amiga 3,5-Zoll-Diskette — DM 69,–*

* Unabhängig von der Anzahl der bestellten Programme berechnen wir für das Inland DM 4,– bzw. für das Ausland DM 6,– Porto und Verpackung. Bitte benutzen Sie die Bestellkarte.

ORGATEC INTERNATIONALE BÜROMESSE KÖLN 25.–30. OKT. 1990
Wir stellen aus HALLE 2.1 GANG F, STAND 42

DMV-Verlag · Postfach 2 50 · 3440 Eschwege

DMV Daten- und Medienverlag